APPLICATION OF THE SAHA EQUATION TO HIGH TEMPERATURE (≥ 6000K) ROCKET EXHAUST

Robert T. Nachtrieb

Phillips Laboratory
OLAC–PL/RKFE
Edwards AFB, California 93524–7680

March 1993

Final Report (Second Edition)

DTIC
ELECTE
APR 15 1993
S E D

93-07814

93   069
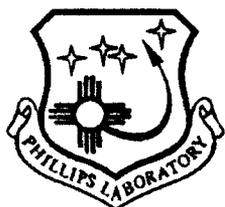
PHILLIPS LABORATORY
Propulsion Directorate
AIR FORCE MATERIEL COMMAND
EDWARDS AIR FORCE BASE CA 93524–5000

NOTICE

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related G overnment procurement operation, the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, or in any way licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may be related thereto.

FOREWORD

This final report was submitted by the OLAC PL/RKFE Branch, at the Phillips Laboratory, Edwards AFB CA 93524−7680. OLAC PL Project Manager was Franklin B. Mead

This report has been reviewed and is approved for release and distribution in accordance with the distribution statement on the cover and on the SF Form 298.

FRANKLIN B. MEAD
Project Manager
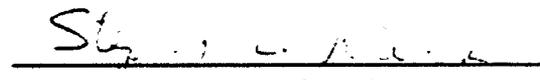
STEPHEN L. RODGERS
Chief, Emerging Technologies Branch
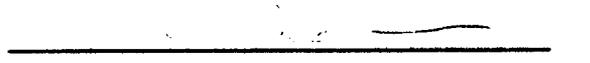
LEONARD C. BROLINE, Lt Col, USAF
Director,
Fundamental Technologies Division
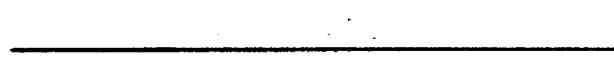
RANNEY G. ADAMS
Public Affairs Director

REPORT DOCUMENTATION PAGE

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE MARCH 1993	3. REPORT TYPE AND DATES COVERED FINAL FEBRUARY 1993

4. TITLE AND SUBTITLE

Application of the SAHA Equation to High Temperature (≥ 6000 K) Rocket Exhaust.

5. FUNDING NUMBERS

PE: 62302F
PR: 3058
TA: 0096

6. AUTHOR(S)

Robert T. Nachtrieb

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Phillips Laboratory
OLAC–PL/RKFE
Edwards AFB, California 93524–7680

8. PERFORMING ORGANIZATION REPORT NUMBER

PL–TR–92–3042R

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

COSATI CODE: 21/08

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved For Public Release, Distribution is Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Using the SAHA equations and spectroscopic constants, the computer program in this report calculates the species populations of diatomic molecules, neutrals, ions, and electrons in a plasma. The code considers the equilibrium of a two–elements, chemically reacting plasma, by calculating the partition function for each species. By rederiving theoretically, as the JANNAF tables do, the thermodynamic properties of a rocket propellant, but using extended excitation levels, the program can estimate (above 6000 K) performance of advanced propulsion concepts.

This second edition code described here considers in addition to mixtures of carbon (C), hydrogen (H), nitrogen (N), and oxygen (O), the species of argon (Ar), up through C_V, H_{II}, N_{IV}, O_V, and Ar_V; any diatomic combination of these elements, both neutral and singly ionized. The user can enter in spectroscopic data for his own elements. Program results agree with JANNAF tables at 6000 K, for dominant species; output graphs of nitrogen species densities (3000 K to 30,000 K) and air species mole fractions (3000 K to 10,000 K) match published data. The second edition also has the additional feature of being able to calculate rocket performance based upon the input of a specific quantity of energy when running the code. This is especially valuable for calculating the high temperature performance of laser, fusion, and antiproton thermal propulsion systems which add heat to a working fluid or propellant.

14. SUBJECT TERMS

SAHA Equations; Interplanetary Missions; Spectroscopy Constants
Thermodynamic Properties of Rocket Propellants; Fortran Code

15. NUMBER OF PAGES

157

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR

Contents

List of Tables

List of Figures

Acknowledgments

1. INTRODUCTION

Interplanetary missions will need rockets with high specific impulse and high specific power. Nuclear fusion, the power source of the Sun, could power a rocket for these interplanetary missions. [1]–[8] While a voyage to Mars, for example, would take almost a year with chemical propulsion, a fusion–powered rocket could get there in months. The high temperature of fusion plasmas (already found in experimental fusion reactor concepts) would give I_{SP}'S of 1000's of seconds. The large energy release from fusion reactions would give the necessary specific power.

If the rocket could pick up more propellant when it arrived at Mars, it would not need to carry out propellant for the voyage back. This would reduce the overall mass of the inflight rocket, increase the thrust–to–weight ratio, and reduce transit time. A source of such propellant is the Martian atmosphere, which is 95% CO_2.

The JANAF Thermodynamic Tables [9] give, up to 6000K, the enthalpy, entropy, and heat capacity of several elements and molecules. However, fusion plasmas have temperatures in the 100's of keV (10^9 K). By rederiving theoretically, as the JANAF tables do, the theromodynamic properties of the fusion rocket propellant, but using extended excitation levels, we can estimate rocket performance above 6000 K.

At high temperatures, as molecules in the gas dissociate into their atomic components, chemistry simplifies. Although probability increases with temperature that a molecule will occupy a higher excited state, the number of molecules decreases with temperature. Once most of the molecules have dissociated, the remaining molecules have little effect on the thermodynamic properties of the total gas.

At temperatures below 6000 K, the number of chemicals to consider is overwhelming. Computer codes exist, such as Selph's "ISP Code"[10], that first identify chemicals present

at a given temperature and pressure, and then interpolate JANAF table data to find their partial densities at equilibrium. From this they calculate the total gas thermodynamic properties.

The code described here considers: mixture of argon (Ar), carbon (C), hydrogen (H), nitrogen (N), and oxygen (O), up through Ar_V, C_V, H_{II}, N_{IV}, and O_V; any diatomic combinations of these elements, both neutral and singly ionized. The user can enter in spectroscopic data for his own elements. Program results agree with Selph's code at 6000 K, for dominant species, output graphs of nitrogen species densities (3000 K to 30,000 K) and air species mole fractions (3000 K to 10,000 K) match Cambel[11].

The second edition also has the additional feature of being able to calculate rocket performance based upon the input of a specific quantity of energy when running the code. This is especially valuable for calculating the high temperature performance of laser, fusion, and antiproton thermal propulsion system which add heat to a working fluid or propellant.

2 THEORY

2.1 Saha Equation

Mass Action. Assume diatomic molecules, A_2, which dissociate into atoms, A. At equilibrium, write the dissociation reaction

$$A_2 \rightleftharpoons 2A - u_{D,A_2},$$ (1)

where u_{D,A_2} represents the dissociation energy of the reaction. According to the law of mass action [12], write the equilibrium constant of the reaction in terms of the inventories, N, of the individual species.

$$K_{A_2} = \frac{N_A^2}{N_{A_2}}.$$ (2)

Partition Function. The total energy of a particle consists of its translational and internal energies. Neglect nuclear excitation, and assume separable rotational, vibrational, and electronic energies, to write the particle Hamiltonian, $\mathcal{H}$,

$$\mathcal{H}_{tot} = \mathcal{H}_{tr} + \mathcal{H}_{rot} + \mathcal{H}_{vib} + \mathcal{H}_{el}.$$ (3)

For a particle partition function, write[13]

$$Z_{tot} = Z_{tr} Z_{rot} Z_{vib} Z_{el},$$ (4)

where

$$Z_{tr} = \left(\frac{2\pi m}{h^2}\right)^{3/2} (kT)^{3/2} V,$$ (5)

$$Z_{el} = \sum_i g_i \exp\left(\frac{-u_i}{kT}\right),$$ (6)

$$Z_{rot} \approx \frac{T}{\Theta_{rot}},$$ (7)

$$Z_{vib} \approx \frac{T}{\Theta_{vib}},$$ (8)

and m represents the particle mass; h represents Planck's constant; k represents Boltzmann's constant; T represents the gas temperature; V represents the gas volume; u_i represents the energy of the ith electronic excitation state; $g_i = 2J + 1$ represents the degeneracy (multiplicity) of the ith electronic excitation state; J represents the total angular momentum quantum number; Θ_{rot} represents the characteristic temperature for rotation of the ground state (electronic) of the molecule, and

$$\Theta_{rot} = \frac{8\pi^2 I kT}{h^2},$$ (9)

2

where I represents the moment of inertia of the molecule; Θ_{vib} represents the characteristic temperature for vibration of the ground state (electronic) of the molecule, and

$$\Theta_{vib} = \frac{h\nu}{k}, \tag{10}$$

where ν represents the vibration frequency of the molecule.

Electronic excitation energies, u_i, and the degeneracies, g_i, come from spectroscopic data[14]. Similarly, find Θ_{rot} and Θ_{vib} for diatomic molecules in Huber & Herzberg[15].

Atomic species have no vibrational or rotational energy, so that

$$Z_{rot} = Z_{vib} = 1 \text{ (atomic species)}. \tag{11}$$

In terms of the total partition function, write the reaction equilibrium constant in Eq.(2) as

$$K_{A_2} = \frac{Z_A^2}{Z_{A_2}} \exp\left(\frac{-u_{D,A_2}}{kT}\right). \tag{12}$$

Ideal Gas. Assume an ideal gas (a better approximation as more molecules dissociate),

$$PV = NkT, \tag{13}$$

and substitute V into Eq.(5):

$$Z_{tr} = \left(\frac{2\pi m}{h^2}\right)^{3/2} \frac{(kT)^{5/2}}{P} N, \tag{14}$$

where N represents the total number of particles in the gas, specifically "heavy" particles, N_h, (molecules, atoms, ions) and electrons, N_e.

$$N = N_h + N_e. \tag{15}$$

Normalize all species particle inventories to the total number of nuclei, N_0, present in the gas, and define the nuclear fractions (similar to Cambel[11]):

$$y_{A_2} = \frac{2N_{A_2}}{N_0}, \tag{16}$$

$$y_A = \frac{N_A}{N_0}, \tag{17}$$

$$y_h = \frac{N_h}{N_0}, \text{ and} \tag{18}$$

$$y_e = \frac{N_e}{N_0}. \tag{19}$$

Using Eqs.(16)-(19) in Eqs.(2) and (12) gives the first Saha equation:

$$\frac{y_A^2}{y_{A_2}(y_h + y_e)} = \left(\frac{2\pi}{h^2}\right)^{3/2} \left(\frac{m_A m_A}{m_{A_2}}\right)^{3/2} \frac{(kT)^{5/2}}{P} \frac{Z_{el,A}}{(Z_{el}Z_{rot}Z_{vib})_{A_2}} \exp\left(\frac{-u_{D,A_2}}{kT}\right), \tag{20}$$

3

using the component partition functions for the total partition functions, and simplifying the translational partition function for A_2 and A. Note that by taking $m_{A_2} = m_A + m_A$, the second right-hand-side term (in parentheses) becomes the reduced mass of the particles. Also note that, for a given temperature and pressure, the value of the right-hand-side of Eq.(20) is known (abbreviated R).

Ionization. Now consider ionization as well as dissociation. At equilibrium,

$$A^{+j} \rightleftharpoons A^{+j+1} + e^- - u_{I,j}, \tag{21}$$

where $j + 1$ represents the degree of ionization and $u_{I,j}$ represents the energy of the jth ionization. Equate the two forms of the reaction equilibrium constant to write

$$\frac{y_{A^{+j+1}} y_e}{y_{A^{+j}}(y_h + y_e)} = \left(\frac{2\pi m_e}{h^2}\right)^{3/2} \frac{(kT)^{5/2}}{P} \frac{Z_{el,A^{+j+1}}}{Z_{el,A^{+j}}} \exp\left(\frac{-u_{I,j}}{kT}\right). \tag{22}$$

Chemical Reactions. Now consider a chemical reaction between elements A and B that forms the compound AB. At equilibrium,

$$AB \rightleftharpoons A + B - u_{R,AB}, \tag{23}$$

and so

$$\frac{y_A y_B}{y_{AB}(y_h + y_e)} = \left(\frac{2\pi}{h^2}\right)^{3/2} \left(\frac{m_A m_B}{m_A + m_B}\right)^{3/2} \frac{(kT)^{5/2}}{P} \frac{Z_{el,A} Z_{el,B}}{(Z_{el} Z_{rot} Z_{vib})_{AB}} \exp\left(\frac{-u_{R,AB}}{kT}\right). \tag{24}$$

If the compound AB ionizes,

$$AB \rightleftharpoons AB^+ + e^- - u_{I,AB}, \tag{25}$$

and

$$\frac{y_{AB^+} y_e}{y_{AB}(y_h + y_e)} = \left(\frac{2\pi m_e}{h^2}\right)^{3/2} \frac{(kT)^{5/2}}{P} \frac{(Z_{el} Z_{rot} Z_{vib})_{AB^+}}{(Z_{el} Z_{rot} Z_{vib})_{AB}} \exp\left(\frac{-u_{I,AB}}{kT}\right). \tag{26}$$

Saha Equations. Solve the following set of non-linear, coupled, algebraic equations:

$$\frac{y_{A^+} y_e}{y_A(y_h + y_e)} = R_{A1}, \tag{27}$$

$$\frac{y_{A^{+2}} y_e}{y_{A^+}(y_h + y_e)} = R_{A2}, \tag{28}$$

$$\frac{y_{A^{+3}} y_e}{y_{A^{+2}}(y_h + y_e)} = R_{A3}, \tag{29}$$

$$\frac{y_{A^{+4}} y_e}{y_{A^{+3}}(y_h + y_e)} = R_{A4}, \tag{30}$$

4

$$\frac{y_A^2}{y_{A_2}(y_h + y_e)} = R_{A6}, \tag{31}$$

$$\frac{y_{A_2^+}}{y_{A_2}(y_h + y_e)} = R_{A7}, \tag{32}$$

$$\frac{y_{B^+} y_e}{y_B(y_h + y_e)} = R_{B1}, \tag{33}$$

$$\frac{y_{B^{+2}} y_e}{y_{B^+}(y_h + y_e)} = R_{B2}, \tag{34}$$

$$\frac{y_{B^{+3}} y_e}{y_{B^{+2}}(y_h + y_e)} = R_{B3}, \tag{35}$$

$$\frac{y_{B^{+4}} y_e}{y_{B^{+3}}(y_h + y_e)} = R_{B4}, \tag{36}$$

$$\frac{y_B^2}{y_{B_2}(y_h + y_e)} = R_{B6}, \tag{37}$$

$$\frac{y_{B_2^+}}{y_{B_2}(y_h + y_e)} = R_{B7}, \tag{38}$$

$$\frac{y_A y_B}{y_{AB}(y_h + y_e)} = R_{AB1}, \tag{39}$$

$$\frac{y_{AB^+}}{y_{AB}(y_h + y_e)} = R_{AB2}, \tag{40}$$

$$y_h = y_A + y_{A^+} + y_{A^{+2}} + y_{A^{+3}} + y_{A^{+4}} + y_{A_2} + y_{A_2^+} + \tag{41}$$
$$y_B + y_{B^+} + y_{B^{+2}} + y_{B^{+3}} + y_{B^{+4}} + y_{B_2} + y_{B_2^+} +$$
$$y_{AB} + y_{AB^+},$$

$$y_e = y_{A^+} + 2y_{A^{+2}} + 3y_{A^{+3}} + 4y_{A^{+4}} + y_{A_2^+} + \tag{42}$$
$$y_{B^+} + 2y_{B^{+2}} + 3y_{B^{+3}} + 4y_{B^{+4}} + y_{B_2^+} +$$
$$y_{AB^+},$$

$$y_{0,A} = y_A + y_{A^+} + y_{A^{+2}} + y_{A^{+3}} + y_{A^{+4}} + 2y_{A_2} + 2y_{A_2^+} + y_{AB} + y_{AB^+}, \tag{43}$$

$$y_{0,B} = y_B + y_{B^+} + y_{B^{+2}} + y_{B^{+3}} + y_{B^{+4}} + 2y_{B_2} + 2y_{B_2^+} + y_{AB} + y_{AB^+}, \tag{44}$$

$$1 = y_{0,A} + y_{0,B}, \tag{45}$$

where the R (for "RHS", right-hand-side) represent the known functions of temperature and pressure, and $y_{0,A}$ and $y_{0,B}$ represent the stoichiometric fractions of elements A and B.

With solved Eqs.(27)-(45), the mole fraction of the kth species becomes

$$\frac{N_k}{N} = \frac{N_0 y_k}{N_0(y_h + y_e)} = \frac{y_k}{(y_h + y_e)}. \tag{46}$$

Similarly, the partial density of the kth species becomes

$$\frac{N_k}{V} = N_0 y_k \frac{P}{N_0(y_h + y_e)kT} = \frac{y_k}{(y_h + y_e)} \frac{P}{kT}. \tag{47}$$

5

2.2 Thermodynamic Functions

Sum the energies in translation, excitation, vibration, electronic excitation, dissociation, ionization, and chemical reactions to get the total energy, U, of the gas. Thus,

$$U = U_{tr} + U_{rot} + U_{vib} + U_{el} + U_D + U_I + U_R. \tag{48}$$

Sum the internal energy of the gas and the thermal excitation to get the enthalpy of the gas, H:

$$H = U + PV. \tag{49}$$

Divide H by $n = N/N_{Av}$, the number of moles, where N_{Av} is Avagadro's number, to get the molar enthalpy, h:

$$
\begin{aligned}
h &= H\frac{N_{Av}}{N}, \tag{50} \\
&= N_{Av}\frac{U}{N} + N_{Av}kT,
\end{aligned}
$$

substiting $PV = NkT$ into the second term. Divide the gas energy by the number of particles to get the average particle energy,

$$
\begin{aligned}
\bar{u} &= \frac{U}{N}, \tag{51} \\
&= \bar{u}_{tr} + \bar{u}_{vib} + \bar{u}_{rot} + \bar{u}_{el} + \bar{u}_D + \bar{u}_I + \bar{u}_R.
\end{aligned}
$$

Examine each of the average energies separately.

Each of the three degrees of freedom of translation has $kT/2$, and so the average translational energy, $\bar{u}_{tr}$, becomes

$$\bar{u}_{tr} = \frac{3}{2}kT. \tag{52}$$

Assume that the average rotational and vibrational energies for diatomic molecules have kT each, a valid approximation at high temperatures.

$$\bar{u}_{rot} = \bar{u}_{vib} = kT. \tag{53}$$

Find the average electronic excitation energy, $\bar{u}_{el}$, from

$$
\begin{aligned}
\bar{u}_{el} &= \frac{\sum_i u_i g_i \exp(-u_i/kT)}{\sum_i g_i \exp(-u_i/kT)}, \tag{54} \\
&= \frac{\sum_i u_i g_i \exp(-u_i/kT)}{Z_{el}}.
\end{aligned}
$$

Find the average dissociation, ionization, and reaction energies:

$$
\begin{aligned}
\bar{u}_D &= u_D, \tag{55} \\
\bar{u}_I &= u_I, \text{ and} \tag{56} \\
\bar{u}_R &= u_R. \tag{57}
\end{aligned}
$$

Recognize that $N_{Av}k = R = 8.314$ J/mol, the gas constant, so the enthalpy becomes

$$h = RT \left(\frac{5}{2} + (2) + \frac{\bar{u}_{el} + u_D + u_I - (u_R)}{kT} \right). \tag{58}$$

Atomic species do not have the inner terms in parentheses, since they have neither rotational nor vibration energies, nor have reacted chemically to form a compound.

Define the constant pressure heat capacity, c_p,

$$
\begin{aligned}
c_p &= \frac{\partial}{\partial T} h, \tag{59} \\
&= R \left(\frac{5}{2} + (2) \right) + N_{Av} \frac{\partial}{\partial T} \bar{u}_{el}.
\end{aligned}
$$

Recall the definition of $\bar{u}_{el}$ from Eq.(55), to find the rightmost term:

$$
\begin{aligned}
\frac{\partial}{\partial T} \bar{u}_{el} &= \frac{\partial}{\partial T} \left(\frac{\sum_i u_i g_i \exp(-u_i/kT)}{Z_{el}} \right), \tag{60} \\
&= \frac{1}{Z_{el}} \frac{\partial}{\partial T} \sum_i u_i g_i \exp(-u_i/kT) - \\
&\quad \frac{\sum_i u_i g_i \exp(-u_i/kT)}{Z_{el}^2} \frac{\partial}{\partial T} Z_{el}, \\
&= \frac{1}{kT^2} \left(\frac{\sum_i u_i^2 g_i \exp(-u_i/kT)}{Z_{el}} - \right. \\
&\quad \left. \frac{\sum_i u_i g_i \exp(-u_i/kT)}{Z_{el}} \frac{\sum_i u_i g_i \exp(-u_i/kT)}{Z_{el}} \right), \\
&= \frac{1}{kT^2} \left(\overline{u_{el}^2} - \bar{u}_{el}^2 \right).
\end{aligned}
$$

Therefore, write the heat capacity

$$c_p = R \left(\frac{5}{2} + (2) + \frac{\overline{u_{el}^2} - \bar{u}_{el}^2}{kT^2} \right). \tag{61}$$

Express a difference in entropy by[16]

$$s_2 - s_1 = \int_{T_1}^{T_2} c_p \frac{dT}{T} - R \ln \frac{P_2}{P_1}. \tag{62}$$

Operate on c_p, in Eq.(61), to evaluate the first right-hand-side term:

$$\int_{T_1}^{T_2} c_p \frac{dT}{T} = R \left(\frac{5}{2} + (2) \right) \ln \frac{T_2}{T_1} + N_{Av} \int_{T_1}^{T_2} c_p \frac{\partial}{\partial T} \bar{u}_{el} \frac{dT}{T}, \tag{63}$$

$$\int_{T_1}^{T_2} c_p \frac{\partial}{\partial T} \bar{u}_{el} \frac{dT}{T} = \int_{u_1}^{u_2} \frac{d}{dT} \bar{u}_{el},\tag{64}$$

$$= \frac{\bar{u}_{el}}{T} - \int_{T_1}^{T_2} \bar{u}_{el} d\left(\frac{1}{T}\right).$$

Note that

$$\frac{\partial}{\partial(1/T)} Z_{el} = \frac{\partial}{\partial(1/T)} \sum_i g_i \exp(-u_i/kT)m,\tag{65}$$

$$= \sum_i g_i \exp(-u_i/kT)\left(\frac{-u_i}{k}\right)m,$$

$$= \frac{-1}{k} \sum_i u_i g_i \exp(-u_i/kT)m,$$

$$= \frac{-\bar{u}_{el} Z_{el}}{k}.$$

So

$$\int \bar{u}_{el} d\left(\frac{1}{T}\right) = -k \int \frac{1}{Z_{el}} \frac{\partial}{\partial(1/T)} Z_{el} d\left(\frac{1}{T}\right),\tag{66}$$

$$= -k \int \frac{dZ_{el}}{Z_{el}}.$$

Substitute all the terms back into the original equation to get

$$s_2 - s_1 = R\left[\left(\frac{5}{2} + (2)\right) \ln \frac{T_2}{T_1} + \frac{\bar{u}_{el}}{kT} + \ln \frac{Z_{el,2}}{Z_{el,1}} - \ln \frac{P_2}{P_1}\right].\tag{67}$$

For convenience, take state 1 at standard temperature and pressure (STP: $T = 298.15$ K, $P = 1$ atm).

Gas Thermodynamic Properties. Equations (58),(61), and (67) give formulae for the enthalpy, heat capacity, and entropy difference for individual species present in the gas. To get the entropy, heat capacity, and entropy for the entire gas, sum the contributions from each species. For example, write the gas enthalpy

$$h_{gas} = \sum_i h_i \frac{y_i}{(y_h + y_e)},\tag{68}$$

where h_i represents the enthalpy of the ith species, and $y_i/(y_h + y_e)$ represents the mole fraction of the ith species. Find gas heat capacity and enthalpy difference in a similar manner.

Absorbed Thermal Power To calculate the maximum absorbed thermal power, assume that the working fluid (propellant) initially has no enthalpy, so $h_0 = 0$. The maximum absorbed thermal power, P_{abs}, becomes

$$P_{abs} = \dot{m} h_{gas},\tag{69}$$

where $\dot{m}$ represents the mass flow rate of the propellant.

8

2.3 Rocket Applications

To estimate rocket performance, make several assumptions [18]:

1. Homogeneous propellant does not vary in composition throughout the rocket chamber and nozzle.

2. The propellant obeys the perfect gas laws.

3. The propellant flows adiabatically.

4. The propellant flows steadily; expansion occurs isentropically, without shocks.

5. Exhaust gases have axially directed velocity.

6. Nozzle expansion does not change the chemical equilibrium established in the chamber ("frozen flow").

7. Propellant contains gaseous species only.

The diatomic species do not exacly follow the ideal gas assumptions, since they have degrees of freedom in rotation and vibration. At high temperatures, as the diatomic species dissociate, the ideal gas assumption improves. As the nozzle expands and the temperature and pressure decrease, the dissociation/ionization equilibrium changes. However, assume that expansion occurs quickly enough that the equilibrium at the exit remains unchanged from the chamber, and that the expansion occurs isentropically (reversibly). Use the frozen flow approximation, rather than calculate the equilibrium conditions at both the chamber and exit, since the Saha equations do not give correct answers at very low pressures.[1]

Since the expansion occurs isentropically,

$$\frac{T_{ex}}{T} = \left(\frac{P_{ex}}{P}\right)^{\frac{\gamma-1}{\gamma}},$$ (70)

where $\gamma = 5/3 = c_p/c_v$ for an ideal gas. The heat capacity for an ideal gas remains constant at $c_p = R5/2$, so conservation of energy yields

$$\frac{1}{2}(v_{ex}^2 - v^2) = h_{ex} - h,$$ (71)
$$= c_p(T_{ex} - T).$$

Assume that the axial velocity of the propellant in the chamber, v, is small compared to the exit velocity, v_{ex}, to solve for the exit velocity:

$$v_{ex} = \sqrt{2c_p(T_{ex} - T)},$$ (72)

---

[1]The Saha equations describe equilibrium brought about from collisions. At low pressures, fewer collisions occur, and equilbrium occurs by way of longer-range interactions ("coronal equilibrium"). This project considered only Saha equilibrium.

$$= \sqrt{5RT\left[1 - \frac{T_{ex}}{T}\right]},$$

$$= \sqrt{5RT\left[1 - \left(\frac{P_{ex}}{P}\right)^{2/5}\right]}.$$

Convert R from J/K/mol to J/K/kg by dividing R by the average molecular weight, $\overline{MW}$, where

$$\overline{MW} = \frac{y_{0,A}MW_A + y_{0,B}MW_B}{(y_h + y_e)}, \tag{73}$$

to get

$$v_{ex} = \sqrt{5\frac{R}{\overline{MW}}T\left[1 - \left(\frac{P_{ex}}{P}\right)^{2/5}\right]}. \tag{74}$$

The exit velocity gives thrust, F, specific impulse, I_{sp}, and jet power, P_{jet}:

$$F = \dot{m}v_{ex}, \tag{75}$$

$$I_{sp} = \frac{v_{ex}}{g}, \text{ and} \tag{76}$$

$$P_{jet} = \frac{1}{2}\dot{m}v_{ex}^2. \tag{77}$$

3 USER'S MANUAL

This section tells how to use the code, what the input parameters mean, what techniques work best for the desired results, where to look for output, and how to create new input elements. If the user has trouble, the author receives internet email at the address nachtrieb@uiuc.edu, or US mail care of the Phillips Laboratory.

3.1 Starting Up

By making a directory on the hard-drive to which all the program files may be copied, the user can keep input and output files separate from other programs, and can keep track of the output files the program creates. To do so, at the prompt (something like C:\>), type:

```
C:\> md saha
C:\> copy a:*.* c:\saha
C:\> cd \saha
C:\SAHA>
```

These commands make a directory named SAHA, copy the program files from floppy to the new directory, and then change into the new directory.

To run the code, type the program name, ab:

```
C:\SAHA> ab
```

Several examples might prove helpful for demonstrating how to operate the code.

3.2 Example: Carbon Dioxide, One Temperature

A few lines of introduction greet the user, after which the program prompts the user to indicate which two elements he wishes the program to work with.

```
Program: AB.FOR, ver. 1.3; Author: R.Nachtrieb, Jan 93;
OLAC-PL/RKFE Edwards AFB CA 93523-5000
internet email: nachtrieb@uiuc.edu



Ar, C, H, N, O
Which two (2) elements to use?
[C ,O ]
```

The user types the letters corresponding to the symbols of the elements he wishes to consider. The program suggests four elements: carbon, hydrogen, nitrogen, and oxygen, as indicated by C, H, N, O. In hard brackets, the program displays the current (default) setting. If the user wants to use the default settings, he simply presses <Enter>. Otherwise,

11

he can type in two element symbols, separated by either a space or a comma. Assume that the user accepts the default elements and presses <Enter>.

Once the user has entered in his selections, the program echoes back the selection. Note that the program is case sensitive, so, for example, it does not recognize n as nitrogen; the user should use N for nitrogen.

```
Using C ,O ,CO
```

```
Stoichiometric Proportions of C ,O ?
[  1.000E+00,  2.000E+00]
```

The program then asks the user for the relative proportions of each element. For example, if the user wished to analyze carbon dioxide, the user could indicate that there was one part of carbon for two parts of oxygen. Note that this program does *not* consider polyatomic molecules, only diatomic; but the user can approximate gases like CO_2 by giving the same stoichiometric proportions as CO_2.

```
Proportions of C ,O   1.000E+00  2.000E+00
```

```
Chamber, exit pressures (atm)?
[  1.000E+00,  0.000E+00]
```

The program prompts the user for the chamber pressure, in atmospheres, to use in the Saha equations. The exit pressure is used to calculate ideal rocket performance. At low pressures, the Saha equations no longer give the correct equilibrium species distribution, since collisions between gas molecules occur infrequently. Thus, while the program may give answers for chamber pressures below 10^{-3} atm, the user should not trust them. The program does not invoke the Saha equations to calculate equilibrium conditions at the nozzle exit, so the user may enter zero pressure (vacuum) for the exit.

```
press1,press2 (atm)   1.000E+00  0.000E+00
```

```
Specify Temperature (T) or Absorbed Power (P)?
[T]
```

The program asks the user either to specify the chamber temperature and calculate the absorbed power, or to specify the absorbed power and calculate the chamber temperature. The program actually uses the chamber temperature as an independent variable and calculates the absorbed power. When the users specifies a desired absorbed power, the program varies the chamber temperature until the calculated absorbed power matches the desired absorbed power. A later example describes specifying the absorbed power. For this example, choose to specify the chamber temperature, T.

T

```
Chamber temperature range: temp1(K),temp2(K),howmany?
[  6.000E+03,  3.000E+04,    1]
```

The program asks the user for the temperature range to consider. If the user wishes to consider a single temperature, he types in the low and high temperatures of the temperature range, and then types a 1. The program checks to see how many temperatures the user wants to consider: if the user has entered in a 1, the program ignores the second (high) temperature and considers only the first. Thus, to consider a gas with temperature 6000 K, the user types 6000, followed by a space, another temperature (any number will do, since the program ignores it), and then a 1.

The program can run for temperatures below 6000 K. However, since it does not consider polyatomic molecules, the program looses accuracy the further below 6000 K the user specifies the temperature. Below 6000 K, the user should use Selph's "Isp" code.

```
temp1(K),temp2(K),howmany  6.000E+03  3.000E+04    1
```


```
View loop progress: inner,middle,outer?
[  20,  20,  20]
```

The numbers that the user enters tell the program how many loop iterations the program should count before printing a progress message to the screen. On personal computers, the Saha equations may take a long time to converge, especially if the computer does not have a numeric co-processor. To reassure the user that the program runs properly, the program prints convergance factors to the screen, with the number of times the program loops has iterated. Using fast computers, printing out to the screen every loop slows down the program; but using slow computers, the user can't tell if the program works or not, since the program takes a long time to show signs of activity.

Sugestions.   The first time the user runs the program, try entering 20 20 20. If the program seems to iterate the numbers as fast as it can, that is, the program does not pause between progress updates, try increasing the numbers to, say, 100 100 20. The first two numbers need not be greater than 100, since the loops usually converge in fewer than 100 interations.

The program is "done" when the final residual (sum of all the absolute values of the differences between the left-hand-sides and the right-hand-sides of the Saha equations) drops below ϵ, where $\epsilon = 10^{-7}$. When the outer loop residual gets down around 10^{-6}, it starts bouncing around. If the residual doesn't drop below ϵ within a specified number of outer loop iterations (the last of the three numbers the user types), the program stops. Usually 20 outer loop iterations suffices.

Note that only the final of the three numbers affects the accuracy of the program results. In all cases, the loop residuals must drop below ϵ. However, if the program can't quite get the outer loop residual below ϵ, but comes close, the user can stop calculations after, say, 20 iterations.

13

```
seein,seemid,maxout  20  20  20
```

```
mass flow rate (kg/s)?
[  1.000E+00]
```

Finally, the program asks the user for the mass flow rate. The program uses the mass flow rate to determine the thrust an ideal rocket nozzle produces.

After reading the mass flow rate, the program has all the information it needs to run a single temperature case. It echoes back the information, and then starts to work. First it reads in spectroscopic data from disk.

```
mdot:    1
```

```
Getting data from file C_0.in
Getting data from file C_1.in
Getting data from file C_2.in
Getting data from file C_3.in
Getting data from file C_4.in
Getting data from file O_0.in
Getting data from file O_1.in
Getting data from file O_2.in
Getting data from file O_3.in
Getting data from file O_4.in
```

Then the program starts to work on solving the Saha equations. It writes to the screen what temperature (in K) and pressure (now in pascals, P: it considers. It also writes six columns of numbers, which tell the number of loop iterations and the residuals for the three program loops.

```
temp(K), press1(Pa)   6.000E+03   1.013E+05
    in.loop   wyhX-y0X   mid.loop      wyh-1   out.loop   residual
                                               2.000E+01   2.546E-07
```

When the program has finished with a temperature, it prints out the maximum absorbed thermal power and the jet power, thrust, and specific impulse that an ideal rocket would produce. The program also tells which file to examine to get the complete information it calculated.

```
Pabs, Pjet (W)   2.541E+06   5.556E+06
Thrust(N),Isp(s)   3.333E+03   3.399E+02
```

```
Look for output in file (CO.out      )
Press <Enter> to exit program...
```

14

Note that for the last example, the jet power exceeds the maximum absorbed thermal power. This counterintuitive result comes from assumptions made for the ideal rocket. At higher chamber temperatures the maximum absorbed thermal power exceeds the jet power, as it should.

Single-Temperature Output File. We can print a hardcopy of the output file, CO.out, by typing at the MS-DOS prompt:

```
C:\SAHA> type co.out > prn
```

However, after several runs, this may amount to a large stack of paper. An easy way to view the program on the screen uses the "Browse" function of Frailey's DCOM program (or an ASCII text editor).

The first few lines of the output file echo back the users input.

```
Using C ,O ,CO
Stoichiometric Proportions of C ,O
 [  3.333E-01,   6.667E-01]
chamber, exit pressures (atm)
 [  1.013E+05,   0.000E+00]
Temperature range: temp1(K),temp2(K),howmany
 [  6.000E+03,   3.000E+04,     1]
mass flow rate (kg/s)
 [  1.000E+00]
```

Next, the file contains the partition functions for all species considered in the program. Partition functions for atomic species appear in the first five columns, in increasing order of ionization. The sixth and seventh columns contain partition functions for homonuclear diatomic species, also in order of increasing charge. The three rows of data contain the electronic (labeled el at the far left), rotational (rot), and vibrational (vib) partition functions, respectively. (Note that the rotational and vibrational partition functions for the atomic species have 1.000E+00.)

Partition function data for the heteronuclear, diatomic molecule follow the elemental partition function data. Only the diatomic columns contain data, in the same electronic, rotational, and vibrational row order.

```
Partition Function: C
```

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el	9.377E+00	5.939E+00	1.000E+00	2.000E+00	1.000E+00	8.794E+01	5.570E+01
vib	1.000E+00	1.000E+00	1.000E+00	1.000E+00	1.000E+00	2.292E+03	2.515E+03
rot	1.000E+00	1.000E+00	1.000E+00	1.000E+00	1.000E+00	2.249E+00	3.090E+00

15

Partition Function: O

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el	8.947E+00	4.016E+00	8.604E+00	5.646E+00	1.000E+00	8.005E+01	3.594E+01
vib	1.000E+00	1.000E+00	1.000E+00	1.000E+00	1.000E+00	2.886E+03	2.466E+03
rot	1.000E+00	1.000E+00	1.000E+00	1.000E+00	1.000E+00	2.640E+00	2.190E+00

Partition Function: CO

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el						8.390E+01	4.540E+01
vib						3.227E+01	1.073E+02
rot						2.844E+00	3.185E+03

The output file contains the right-hand-sides of the Saha equations, found in Eqs.(27)-(45).

Right-hand-sides of Saha Equation

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
C	1.998E-07	5.122E-19	0.000E+00	0.000E+00	0.000E+00	1.240E+00	6.017E-08
O	1.502E-09	5.827E-27	0.000E+00	0.000E+00	0.000E+00	1.083E+01	2.434E-08
CO						1.782E-04	8.278E-10

The solution to the Saha equations gives the nuclear fractions, that is, the species inventory divided by the total number of nuclei.

Nuclear fractions,y

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
C	1.265E-04	6.198E-07	7.789E-21	0.000E+00	0.000E+00	9.868E-09	2.914E-11
O	3.068E-01	1.131E-05	1.616E-27	0.000E+00	0.000E+00	1.331E-02	7.944E-06
CO						3.332E-01	6.767E-06

yh,ye,yh+ye,beta 6.535E-01 2.664E-05 6.535E-01 4.076E-05

Dividing the nuclear fractions by the sum of the electron fraction and the heavy fraction, yh+ye, gives the mole fractions.

Mole fractions,y/(yh+ye)

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
C	1.935E-04	9.484E-07	1.192E-20	0.000E+00	0.000E+00	1.510E-08	4.458E-11
O	4.695E-01	1.730E-05	2.473E-27	0.000E+00	0.000E+00	2.036E-02	1.216E-05
CO						5.099E-01	1.036E-05
e-	4.076E-05						

The output file contains the densities of the species, per cubic meter.

```
Species densities (m^-3)
      atomic,    atomic,    atomic,    atomic,    atomic,    diatomic,  diatomic,
      neutral    +1         +2         +3         +4         neutral    +1
C     2.367E+20  1.16CE+18  1.458E+04  0.000E+00  0.000E+00  1.847E+16  5.453E+13
0     5.743E+23  2.117E+19  3.025E-03  0.000E+00  0.000E+00  2.490E+22  1.487E+19
CO                                                           6.236E+23  1.267E+19
e-    4.986E+19
```

One finds the molar heats of formation of the species by adding half the dissociation energy to the ionization energies, less the reaction energy. The output file lists heats of reaction for the species; subsequently, the enthalpies listed include the heats of reaction to give "absolute" enthalpy. The output file also lists the total gas enthalpy.

```
Heat of formation (J/mol)
      atomic,    atomic,    atomic,    atomic,    atomic,    diatomic,  diatomic,
      neutral    +1         +2         +3         +4         neutral    +1
C     2.996E+05  1.299E+06  3.462E+06  7.710E+06  1.343E+07  0.000E+00  1.167E+06
0     2.465E+05  1.455E+06  4.574E+06  9.449E+06  1.632E+07  0.000E+00  1.158E+06
CO                                                           -5.152E+05 8.356E+05
```

```
Enthalpies (J/mol)
      atomic,    atomic,    atomic,    atomic,    atomic,    diatomic,  diatomic,
      neutral    +1         +2         +3         +4         neutral    +1
C     4.305E+05  1.512E+06  3.864E+06  8.483E+06  1.471E+07  2.369E+05  1.399E+06
0     3.745E+05  1.686E+06  5.080E+06  1.038E+07  1.785E+07  2.310E+05  1.387E+06
CO                                                           -2.812E+05 -2.850E+05
e-    1.247E+05
CO    gas  3.727E+04
```

The output file also contains species and gas heat capacities and entropy differences (from STP).

```
Heat capacities (C_p) (J/K/mol)
      atomic,    atomic,    atomic,    atomic,    atomic,    diatomic,  diatomic,
      neutral    +1         +2         +3         +4         neutral    +1
C     2.318E+01  2.084E+01  2.083E+01  2.079E+01  2.079E+01  4.169E+01  3.971E+01
0     2.227E+01  2.223E+01  2.166E+01  2.080E+01  2.079E+01  4.024E+01  4.028E+01
CO                                                           4.100E+01  4.000E+01
e-    2.079E+01
CO    gas  3.219E+01
```

Entropy diff (from STP) (J/K/mol)

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
C	2.318E+01	2.084E+01	2.083E+01	2.079E+01	2.079E+01	4.169E+01	3.971E+01
O	2.227E+01	2.223E+01	2.166E+01	2.080E+01	2.079E+01	4.024E+01	4.028E+01
CO						4.100E+01	4.000E+01
e-	6.240E+01						
CO	gas	9.313E+01					

Finally, the output file lists the rocket performance.

```
Thrust(N),Isp(s)   3.333E+03   3.399E+02
Pabs, Pjet (W)   2.541E+06   5.556E+06
```

The program stores the user's input parameters in file AB.in, and offers the parameters as defaults (in hard brackets) the next time the user runs the program.

3.3   Example: Carbon Dioxide, Multiple Temperatures

To examine multiple temperatures, the user enters in data in much the same way as for a single temperature. Once again, run the program,

```
C:\SAHA> ab
```

and provide the program with the information it requests:

```
Program: AB.FOR, ver. 1.3; Author: R.Nachtrieb, Jan 93;
OLAC-PL/RKFE Edwards AFB CA 93523-5000
internet email: nachtrieb@uiuc.edu


Ar, C, H, N, O
Which two (2) elements to use?
[C ,O ]
Using C ,O ,CO

Stoichiometric Proportions of C ,O ?
[  1.000E+00,  2.000E+00]
Proportions of C ,O   1.000E+00   2.000E+00

Chamber, exit pressures (atm)?
[  1.000E+00,  0.000E+00]
press1,press2 (atm)   1.000E+00   0.000E+00
```

18

```
Specify Temperature (T) or Absorbed Power (P)?
[T]
T
Chamber temperature range: temp1(K),temp2(K),howmany?
[  6.000E+03,  3.000E+04,    1]
```

Now the user, instead of selecting only one temperature, inputs the temperature range he wishes to consider, and the number of divisions within that temperature range. If the user enters in 2 for howmany divisions, the program considers the lower and upper bounds; if the user enters 3, the program considers the lower and upper bounds, and a temperature midway between the two, and so on.

Try entering in

```
6e3 30e3 3
```

The program echoes back the user's selection, and then prompts for a new input:

```
temp1(K),temp2(K),howmany  6.000E+03  3.000E+04     3


Print density (1) or mole fraction (2) ?
[1]
```

When the user selects more than one temperature for the program to consider, the program outputs information in a different way: instead of writing all information to one file, the program write specific information to many files. This makes it easy for the user to import a file of, say, species densities vs. temperature into a graphing program. In this way, the user simply looks at the particular file to see enthalpy, heat capacity, entropy, mole fraction, density, or rocket performance. Unfortunately, MS-DOS limits the number of files a user (actually, the program) can have open at one time. Thus the user must choose whether to make files that contain species densities, or to make files that contain species mole fractions. Note that if the user wants both density and mole fraction, he can run the program twice, and specify density the first time and mole fraction the second time.

The rest of the inputs follow the single temperature example.

```
densmol [1]


View loop progress: inner,middle,outer?
[ 100, 100,  20]
seein,seemid,maxout 100 100  20

mass flow rate (kg/s)?
[  1.000E+00]
mdot:   1
```

After the user has input the information, the program reads atomic spectra data from input files, and starts to work.

```
Getting data from file C_0.in
Getting data from file C_1.in
Getting data from file C_2.in
Getting data from file C_3.in
Getting data from file C_4.in
Getting data from file O_0.in
Getting data from file O_1.in
Getting data from file O_2.in
Getting data from file O_3.in
Getting data from file O_4.in
```

Since the program now considers multiple temperatures, it gives, for each temperature, the pressure, progress updates for the loop calculations, and rocket performance.

```
temp(K), press1(Pa)   6.000E+03   1.013E+05
    in.loop   wyhX-yOX   mid.loop      wyh-1   out.loop   residual
                                              2.000E+01  2.546E-07
Pabs, Pjet (W)    2.541E+06   5.556E+06
Thrust(N),Isp(s)   3.333E+03   3.399E+02


temp(K), press1(Pa)   1.800E+04   1.013E+05
    in.loop   wyhX-yOX   mid.loop      wyh-1   ou .loop   residual
                                              2.000E+01  1.126E-06
Pabs, Pjet (W)    7.476E+07   4.561E+07
Thrust(N),Isp(s)   9.551E+03   9.740E+02


temp(K), press1(Pa)   3.000E+04   1.013E+05
    in.loop   wyhX-yOX   mid.loop      wyh-1   out.loop   residual
                                              2.000E+01  1.859E-06
Pabs, Pjet (W)    1.166E+08   9.779E+07
Thrust(N),Isp(s)   1.398E+04   1.426E+03
```

Finally, the program tells the user which files to examine for the output data. Note that the filenames follow mnemonic convention: to see a file containing maximum absorbed thermal power, jet power, rocket thrust, and specific impulse as a function of chamber temperature, see file CO_rkt.out; for carbon species densities, see file C_den.out, and so on.

```
Look for output in files:
(C_eth.out  )
(O_eth.out  )
```

```
(CO_eth.out  )
(C_Cp.out    )
(O_Cp.out    )
(CO_Cp.out   )
(C_etr.out   )
(O_etr.out   )
(CO_etr.out  )
(C_den.out   )
(O_den.out   )
(CO_den.out  )
(CO_rkt.out  )
Press <Enter> to exit program...
```

Multiple-Temperature Output Files. When the users selects multiple temperatures, the output files created contain data in columns. For example, the file CO_rkt.out contains rocket performance data, as a function of temperature:

```
CO   rocket performance parameters
   temp(K)    Pabs(W)    Pjet(W)    thrust(N)  Isp(s)
   6.000E+03  2.541E+06  5.556E+06  3.333E+03  3.399E+02
   1.800E+04  7.476E+07  4.561E+07  9.551E+03  9.740E+02
   3.000E+04  1.166E+08  9.779E+07  1.398E+04  1.426E+03
```

The program creates three output files for density (or mole fraction), enthalpy, heat capacity, and entropy. Thus, file C_den.out contains the desities of all the carbon atomic species, the homonuclear diatomic carbon species, and the total electron density. Similarly, file O_den.out contains density for oxygen species, and the total electron density. The file CO_den.out contains density of carbon monoxide species, and the total electron density. The program writes the same total electron densities to each file as a safety check: if the user plots all the densities on the same graph, discrepancies in electron density alert the user to an error (for example, plotting a file from a different program run).

3.4 Example: Nitrogen, Multiple Temperatures

Suppose the user wants to consider a single element, say nitrogen. Run the program, and after it asks for the elements to consider,

```
Ar, C, H, N, O
Which two (2) elements to use?
[C ,O ]
```

enter in

```
N N
```

The program echoes back the users selection, and asks for the stoichiometric proportions.

```
Using N ,N ,NN
```

```
Stoichiometric Proportions of N ,N ?
[ 1.000E+00,  2.000E+00]
```

At this point, the user can enter any proportions he desires, since both species are the same. However, a much better way exists:

Hint. When considering only one element, type 0 for one of the proportions. This speeds up the code considerably, since it no longer has to consider "heteronuclear" diatomic species. Try

```
1 0
```

after which the program responds,

```
Proportions of N ,N   0.000E+00  1.000E+00
```

After this, the program functions as before.

3.5 Example: Specified Absorbed Power

Now run the program again, but this time specify a maximum absorbed power instead of a chamber temperature.

```
Ar, C, H, N, 0
Which two (2) elements to use?
[C ,O ]
Using C ,O ,CO
```

```
Stoichiometric Proportions of C ,O ?
[ 1.000E+00,  2.000E+00]
Proportions of C ,O   1.000E+00  2.000E+00
```

```
Chamber, exit pressures (atm)?
[ 1.000E+00,  0.000E+00]
press1,press2 (atm)  1.000E+00  0.000E+00
```

```
Specify Temperature (T) or Absorbed Power (P)?
[T] P
P
```

The program then does not ask which temperature range to examine, but goes directly to the loop progress and mass flow rate inputs.

22

```
View loop progress: inner,middle,outer?
[ 100,  100,   20]
seein,seemid,maxout 100 100   20

mass flow rate (kg/s)?
[  1.000E+00]
mdot:   1
```

The program then gets the input data files and calculates the lower and upper bounds on the maximum absorbed thermal power.

```
Getting data from file C_0.in
Getting data from file C_1.in
Getting data from file C_2.in
Getting data from file C_3.in
Getting data from file C_4.in
Getting data from file O_0.in
Getting data from file O_1.in
Getting data from file O_2.in
Getting data from file O_3.in
Getting data from file O_4.in
Getting lower absorbed-power bound...
                                        2.000E+01   2.546E-07

Getting upper absorbed-power bound...
                                        2.000E+01   1.859E-06

Pmin,Pmax (W)   2.541E+06   1.166E+08
```

The program results have been verified for pressure $P = 1$ atm between temperature $T = 6000$ K and $T = 30000$ K. The program thus takes $T = 6000$ K and $T = 30000$ K as the temperature bounds and assumes valid results within those bounds. Using the chemical composition of the propellant and the mass flow rate, the program calculates the maximum absorbed thermal powers at $T = 6000$ K and $T = 30000$ K and displays them to the user. The program then asks the user for the desired absorbed thermal power, Pwant. The desired thermal power must be between the absorbed thermal powers calculated at the boundry temperatures.

```
What power to absorb? (W) 1e7
   resP        epsP       temp            out.loop   residual
   6.476E+00   1.000E-02  1.800E+04        2.000E+01  1.126E-06
   3.243E+00   1.000E-02  1.200E+04        2.000E+01  6.323E-07
   9.781E-01   1.000E-02  9.000E+03        2.000E+01  2.923E-07
   2.619E-01   1.000E-02  7.500E+03        2.000E+01  3.184E-07
   2.144E-01   1.000E-02  8.250E+03        2.000E+01  2.725E-07
```

```
6.200E-02  1.000E-02  7.875E+03          2.000E+01  1.333E-07
6.580E-02  1.000E-02  8.063E+03          2.000E+01  3.229E-07
6.543E-04  1.000E-02  7.969E+03          2.000E+01  1.179E-07


Pwant,Pabs,temp   1.000E+07   9.993E+06   7.969E+03
Pabs, Pjet (W)    9.993E+06   7.808E+06
Thrust(N),Isp(s)  3.952E+03   4.030E+02
```

The program then searches for the temperature at which the calculated absorbed thermal power matches the desired absorbed thermal power, to within 1%. The program displays progress of decrease in the residuals and the search temperatures. If the user enters a desired absorbed power, Pwant, outside the boundries specified by the absorbed thermal powers at $T = 6000$ K and $T = 30000$ K, the program returns the user to the beginning of the data input.

Since the program iterates several temperatures to match the calculated absorbed thermal power to the desired absorbed thermal power, it asks for only one desired power. However, the absorbed thermal power and the chamber temperature have a one-to-one correspondence. The user can plot the absorbed thermal power and jet power as a function of temperature, as in Fig. 1 and find the temperature corresponding to the desired absorbed thermal power.
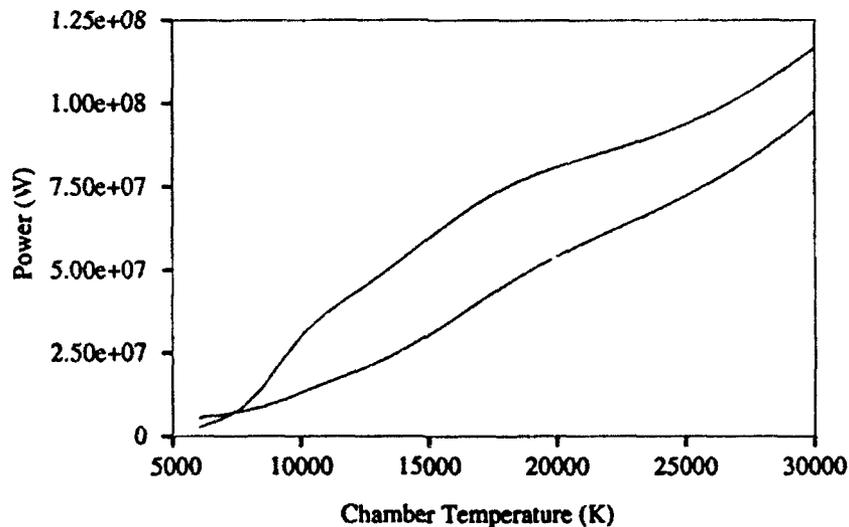
Figure 1: Maximum Absorbed Thermal Power and Ideal Rocket Jet Power vs. Chamber Temperature, CO_2, $P = 1$ atm. Upper curve represents Absorbed Thermal Power.

24

3.6 Example: New Element

Suppose the user wants to run the program with an element the program doesn't yet recognize. The program provides a way for the user to enter the information the program needs to run, and save that information to the input files.

The user runs the program, and enters in the elements he wishes to consider, C and Pn.[2]

```
Ar, C, H, N, O
Which two (2) elements to use?
[N ,N ] C Pn

Oops! Couldn't find Pn in file (ABX.in)
Make a new element? (1=yes)
[1]
```

The file ABX.in contains input information about the elements the program recognizes, and the homonuclear diatomic molecules formed by those elements; specifically, the program looks in ABX.in for the highest ionization state and the mass of the atomic species, for dissociation and ionization energy of the homonuclear diatomic molecule, and for characteristic temperatures of rotation and vibration. The program looks through the input file ABX.in for the element Pn, but doesn't find it. It then asks the user if he wants to define a new element, and provide its characteristic information. If the user says no (0), the program asks the user for two new elements. If the user says yes (1), the program continues.

```
Let's make a new element...

Highest atomic state (1+highest ionization state)?
[0]
```

The highest atomic state refers to the highest ionization level to be considered by the program. State 1 means the program considers only the neutral atom; state 2 means the program considers the neutral atom and the first ionized atom, and so on. Without modification, the program considers up to state 5, four ionizations.

The program then asks for the element molecular weight (in amu), the dissociation and ionization energies of the homonuclear diatomic molecule (in eV). If the diatomic molecule does not form, enter a large, negative dissociation energy, say $u_D = -50$ eV, and a large, postive ionization energy, say $u_I = 50$ eV. This convinces the program the "diatomic" molecule very readily dissociates, i.e. does not form, and cannot ionize (since it does not exist).

```
ist: [0]

Atomic mass (amu)?
```

---

[2]Pandamonium

```
[  1.000E+00]
MW: [  1.000E+00]
```

```
Dissociation energy of diatomic, homonuclear molecule (eV) ?
[ -5.000E+01]
uD: [ -5.000E+01]
```

```
Ionization energy of diatomic, homonuclear molecule (eV) ?
[  5.000E+01]
uI: [  5.000E+01]
```

Next the program asks for the characteristic temperatures of rotation and vibration for the neutral and singly ionized homonuclear diatomic molecule, at the ground electronic state. These can be found in Huber & Herzberg [15] for many diatomic molecules. If no diatomic molecule exists, the characteristic temperatures must still be non-zero, since the program divides by them to get the rotational and vibrational partition functions.

```
Characteristic temperatures, at ground electronicstate (K) ?
Neutral: Rotation, Vibration; Singly Ionized: Rotation, Vibration
[  1.000E+00  1.0C0E+00  1.000E+00  1.000E+00]
```

The program then summarizes the information just typed in about the element and asks if it meets the users satisfaction. If not, the user has the opportunity to re-enter it; if so, the user can append the new element to the file ABX.in, so the element will be "known" by the program the next time the element is used.

```
charT: [  1.000E+00  1.000E+00  1.000E+00  1.000E+00]
```

```
Summary:
Highest atomic state (1+highest ionization state)
[0]
Atomic mass (amu)
[  1.000E+00]
Dissociation energy of diatomic, homonuclear molecule (eV)
[ -5.000E+01]
Ionization energy of diatomic, homonuclear molecule (eV)
[  5.000E+01]
Characteristic temperatures, at ground electronicstate (K)
Neutral: Rotation, Vibration; Singly Ionized: Rotation, Vibration
[  1.000E+00  1.000E+00  1.000E+00  1.000E+00]
Is everything ok? (1=ok,other=redo)
[1]
```

```
Good.  Append data to file (ABX.in)? (0=no,1=yes)
[0]
```

Next, the program searches through the file ABXY.in for the compound formed by the two chosen elements. File ABXY.in contairs energies of dissociation and ionization, and characteristic temperatures for rotation and vibration, for heteronuclear diatomic molecules. If the program cannot find that compound, it prompts the user for the requisite information.

```
Will not append data to file...


Oops! Couldn't find CPn  in file (ABXY.in)
Make a new compound? (1=yes)
[1]


Let's make a new compound...


Reaction energy of diatomic, heteronuclear molecule (eV) ?
[ -5.000E+01]


RAB: [ -5.000E+01]


Ionization energy of diatomic, heteronuclear molecule (eV) ?
[  5.000E+01]


IAB: [  5.000E+01]


Characteristic temperatures, at ground electronicstate (K) ?
Neutral: Rotation, Vibration; Singly Ionized: Rotation, Vibration
[  1.000E+00  1.000E+00  1.000E+00  1.000E+00]
```

As for the homonuclear diatomic molecule, if the two elements form no compound, the user enters a zero reaction energy. The characteristic temperatures for rotation and vibration must be non-zero, though.

The program then summarizes the information, asks if it is satisfactory, and offers to append the new information to the file ABXY.in.

```
TAB: [  1.000E+00  1.000E+00  1.000E+00  1.000E+00]


Reaction energy of diatomic, heteronuclear molecule (eV)
[ -5.000E+01]
Ionization energy of diatomic, heteronuclear molecule (eV)
[  5.000E+01]
Characteristic temperatures, at ground electronicstate (K)
Neutral: Rotation, Vibration; Singly Ionized: Rotation, Vibration
[  1.000E+00  1.000E+00  1.000E+00  1.000E+00]
Is everything ok? (1=ok,other=redo)
```

[1]

```
Good.  Append data to file (ABXY.in)? (0=no,1=yes)
[0]
```

Finally, the program has information it needs about the atomic and molecular species, and so it contiues with the rest of the user input.

```
Will not append data to file...
Using C ,Pn,CPn
```

```
Stoichiometric Proportions of C ,Pn?
[  0.000E+00,  1.000E+00]
```

Atomic Spectra Input. The program reads the atomic spectra data in from files on disk, which must adhere to this naming convention:

```
filename = (element)_(charge).in
```

where (element) represents for the element symbol, and (charge) represents the charge of the atom. For example, if the user wanted to examine Pn up to the fifth ionization state, he would need to enter data into four files, named Pn_0.in through Pn_4.in.

Each file contains data in the following fashion: the first line contains the ionization energy of the atom, in inverse centimeters. Subsequent lines contain pairs of numbers: the degeneracy (dimensionless) and the electronic excitation energy, also in inverse centimeters. The electronic excitation energies increase up to the ionization energy level. The last line contains two negative numbers, to flag the computer to stop reading degeneracy, energy level pairs. The user can find degeneracies and electronic exitation levels for atomic species in Moore[14].

3.7   Benchmarking the Code

Internal Checks. The user can try to fool the code by entering two identical elements with non-zero total nuclear fractions. For example, say the user enters nitrogen and nitrogen.

```
Using N ,N ,NN
Stoichiometric Proportions of N ,N
 [ 5.000E-01,  5.000E-01]
```

The program does not recognize that these two elements are different. Call the two elements N and N'. If the program works properly, the program should predict the same thermodynamic properties for elements N and N', since they really are the same. Also, the program should predict equal quantities of the diatomic molecules N_2, N'_2, and NN'. The output file NN.out, partially listed here, shows that the program predicts correctly.

28

Partition Function: N

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el	4.105E+00	8.937E+00	5.836E+00	1.000E+00	0.000E+00	1.685E+01	3.669E+01
vib	1.000E+00	1.000E+00	1.000E+00	1.000E+00	0.000E+00	2.087E+03	2.159E+03
rot	1.000E+00	1.000E+00	1.000E+00	1.000E+00	0.000E+00	1.768E+00	1.890E+00

Partition Function: N

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el	4.105E+00	8.937E+00	5.836E+00	1.000E+00	0.000E+00	1.685E+01	3.669E+01
vib	1.000E+00	1.000E+00	1.000E+00	1.000E+00	0.000E+00	2.087E+03	2.159E+03
rot	1.000E+00	1.000E+00	1.000E+00	1.000E+00	0.000E+00	1.768E+00	1.890E+00

Partition Function: NN

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
el						1.685E+01	3.669E+01
vib						2.408E+01	1.037E+02
rot						2.779E+00	3.175E+03

Right-hand-sides of Saha Equation

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
N	1.204E-09	8.032E-23	0.000E+00	0.000E+00	0.000E+00	2.319E-03	2.114E-10
N	1.204E-09	8.032E-23	0.000E+00	0.000E+00	0.J00E+00	2.319E-03	2.114E-10
NN						2.319E-03	2.114E-10

Nuclear fractions,y

	atomic, neutral	atomic, +1	atomic, +2	atomic, +3	atomic, +4	diatomic, neutral	diatomic, +1
N	1.390E-02	1.027E-06	5.069E-24	0.000E+00	0.000E+00	1.620E-01	2.104E-06
N	1.390E-02	1.027E-06	5.069E-24	0.000E+00	0.000E+00	1.620E-01	2.104E-06
NN						1.620E-01	2.104E-06

yh,ye,yh+ye,beta 5.139E-01 8.367E-06 5.139E-01 1.628E-05

Nitrogen Density Compare the density vs. temperature profiles of nitrogen, at one atmosphere pressure, in Figures (2) and (3)[11, p.138] They follow each other closely, except Cambel does not consider the N_2^+ species.

Air Mole Fraction Compare the mole fraction vs. temperature profiles of air (80% nitrogen, 20% oxygen), at one atmosphere pressure, in Figures (4) and (5)[11, p.86] Cambel
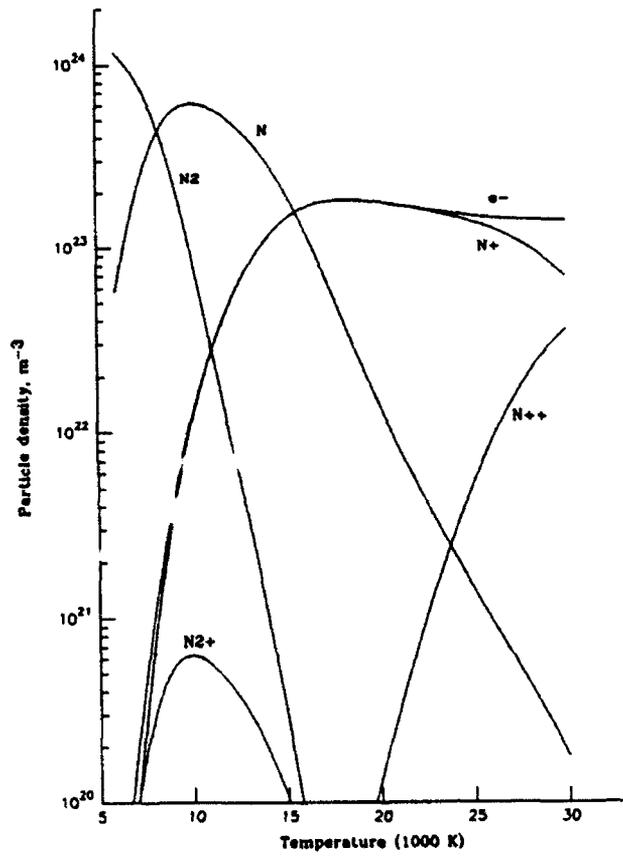
Figure 2: Equilibrium chemical composition of nitrogen plasma at atmospheric pressure.
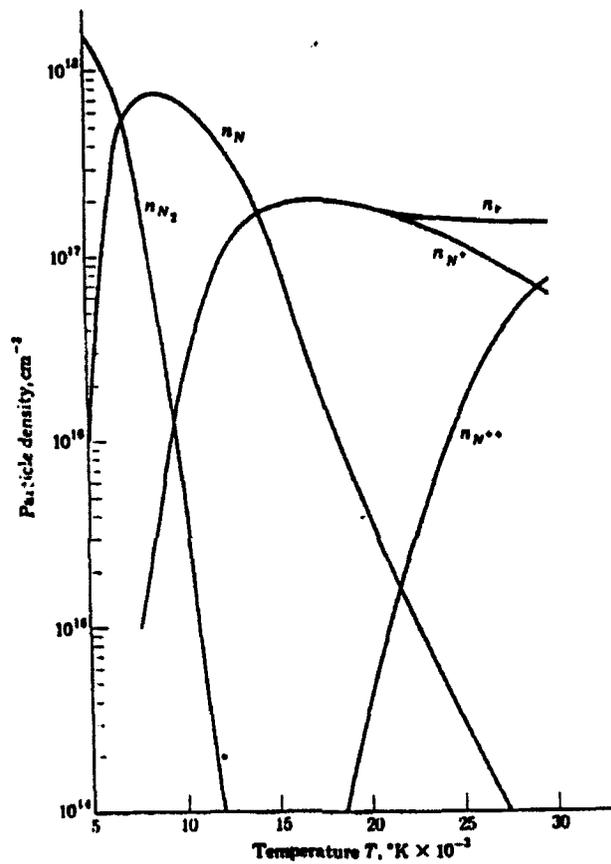
Figure 3: Equilibrium chemical composition of nitrogen plasma at atmospheric pressure. (Cambel, p.138, used without permission)

considers species that this program does not: N_2O, N_2O, O^-. These species effect the electron density, so the program results for electron density do not match well with Cambel, even at 10,000 K. The dominant species at 10,000 K, N, O, and N_2, match within 10% or so. However, Cambel keeps the density constant at the sea-level value, while the program keeps the pressure constant at one atmosphere.
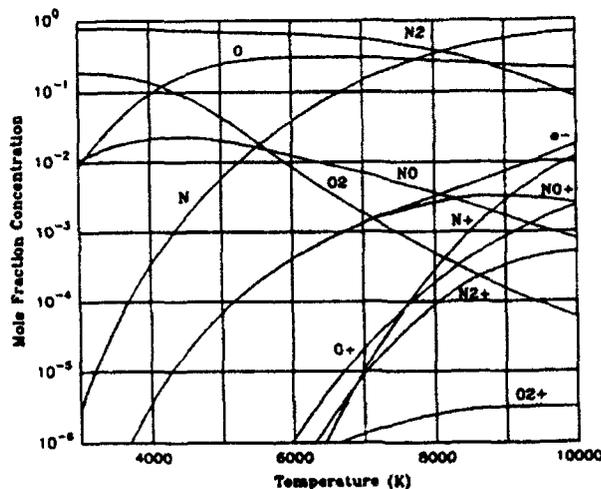
Figure 4: Mole fraction equilibrium chemical composition of air at one atmosphere pressure.
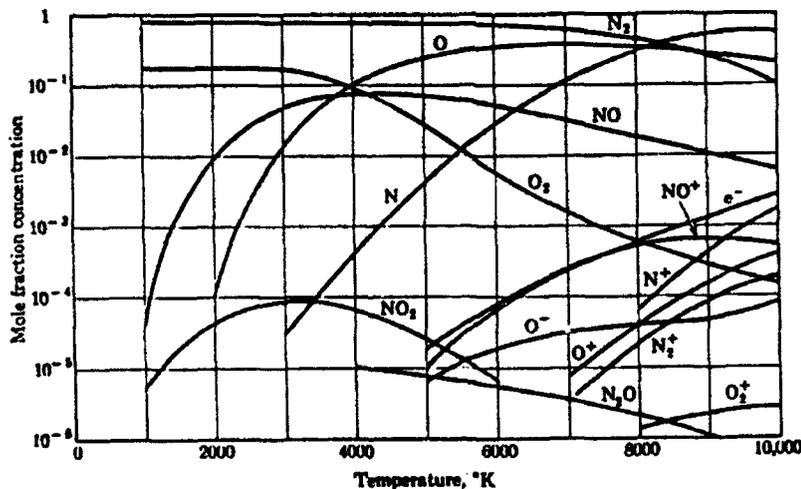
Figure 5: Mole fraction equilibrium chemical composition of air at sea-level density.(Cambel, p.86, used without permission)

Carbon Dioxide The user should use Selph's "Isp" Code for chamber temperatures below 6000 K. Selph's code considers all chemical reactions and all relevant species. However, the JANNAF tables stop at 6000 K, so at temperatures above 6000 K, Selph holds the heat

32

capacity of the gas constant and extrapolates. The author's program calculates theoretically the equilibrium state function using the Saha Equation, but neglects polyatomic molecules. At temperatures much above 6000 K, this assumption becomes minor, as almost all of the molecules have dissociated.

Table (1) compares the dominant species densities predicted by Selph's and the author's codes, for one part carbon and two parts oxygen, at 6000 K chamber temperature and one atmosphere chamber pressure pressure. Note that the dominant species are close, but discrepancies arise to give around 10% difference (for example, atomic carbon, diatomic oxygen, and carbon dioxide). Also note that at 6000 K, there Selph predicts less than 1% CO_2.

Table 1: Particle density comparision between Selph's and author's codes for CO_2, at 6000 K and one atmosphere.

	Selph, "Isp" (m^{-3})	Author, "AB" (m^{-3})
O	6.1E+23	5.7E+23
CO	5.9E+23	6.2E+23
C	1.0E+22	2.3E+20
O_2	8.0E+20	2.4E+22
CO_2	2.2E+20	0.0E+00

Thermodynamic Functions   Table 2 compares thermodynamic properties for nitrogen species N, N^+, and N_2, using the JANNAF tables and the author's code. Results match closely for atomic species, but differ slightly for diatomic species.

33

Table 2: Thermodynamic properties comparison between JANNAF tables and author's program, for some nitrogen species.

		JANNAF	Author
N	$c_p, T = 3000$ K (J/mol/K)	20.98	20.96
	$c_p, T = 6000$ K (J/mol/K)	25.54	25.52
	Δs, 6000 K, 3000 K (J/mol)	15.63	15.62
	Δh, 6000 K, 3000 K (kJ/mol)	68.42	68.40
N+	$c_p, T = 3000$ K (J/mol/K)	20.97	20.96
	$c_p, T = 6000$ K (J/mol/K)	22.39	22.37
	Δs, 6000 K, 3000 K (J/mol)	14.95	14.93
	Δh, 6000 K, 3000 K (kJ/mol)	64.95	65.00
N2	$c_p, T = 3000$ K (J/mol/K)	37.05	37.74
	$c_p, T = 6000$ K (J/mol/K)	38.30	43.42
	Δs, 6000 K, 3000 K (J/mol)	26.11	28.38
	Δh, 6000 K, 3000 K (kJ/mol)	113.20	124.20

References

[1] Bussard, R.W., and DeLauer, R.D., Nuclear Rocket Propulsion, McGraw-Hill, 1958

[2] Chapman, R., Miley, G.H., Kernbichler, W., and Heindler, M., "Fusion Space Propulsion with a Field Reversed Configuration", *Fusion Technology* 15 (1989), Lagrange: American Nuclear Society, pp. 1154-1159

[3] Haloulakos, V.E., and Bourque, R.F., *Fusion Propulsion Study*. AL-TR-89-005, Edwards Air Force Base: Astronautics Laboratory Technical Services Office, 1989

[4] Reinmann, John J., "Fusion Rocket Concepts", *Advanced Propulsion Concepts* (Sixth Symposium, Niagra Falls NY, May 4-6, 1971), National Aeronautics and Space Administration Technical Memorandum X-67826

[5] Yang, T.F., Miller, R.H., Wenzel, K.W., Krueger, W.A., Chang, F.R., "A Tandem Mirror Plasma Source for a Hybrid Plume Plasma Propulsion Concept", Electric Propulsion (18th AIAA/DGLR/JSASS International Conference, Alexandria VA, Sep.30-Oct.2, 1985)

[6] Bussard, Robert W., "Fusion as Electric Propulsion", *Journal of Propulsion and Power.* Vol. 6, No. 5, Sep.-Oct. 1990, Washington D.C.: American Institute of Aeronautics and Astronautics, pp.567-574

[7] Roth, J. Reece, "Space Applications of Fusion Energy", *Fusion Technology.* Vol. 15 (1989), Lagrange: American Nuclear Society, pp. 1375-1394

[8] Sargent, M.G., "A Comparision of Magnetic Confinement Fusion (MCF) and Intertial Confinement Fusion (ICF) for Spacecraft Propulsion", JPL D-5878 Oct. 1988, Pasadena: Jet Propulsion Laboratory

[9] JANAF Thermochemical Tables, The Dow Chemical Company, Midland, MI, 1960,1961,1963,1966,1977

[10] Selph, C., personal conversations, Phillips Laboratory, Jun-Aug 1992

[11] Cambel, Ali Bulent, *Plasma Physics and Magnetofluid-Mechanics*, New York: McGraw Hill, 1963?

[12] Barrow, Gordon M., *Physical Chemistry*, New York: McGraw Hill, 1966

[13] McQuarrie, D.A., *Statistical Mechanics*, New York: Harper & Row, 1976

[14] Moore, Charlotte E., *Atomic Energy Levels*, Washington DC: NSRDS-NBS 35, 1971

[15] Huber, K.P., and Herzberg, G., *Spectra of Diatomic Molecules*, New York: Van Nostrand Reinhold, 1979

[16] Vincenti, Walter G., and Kruger, Charles H. Jr., *Introduction to Physical Gas Dynamics*, New York: Krieger Publishing, 1975

[17] Zemansky, Mark W., *Heat and Thermodynamics*, New York: McGraw Hill, 1957

[18] Sutton, George P., Rocket Propulsion Elements, New York: John Wiley and Sons, 1963, pp. 37-83

A FORTRAN Source Code

A.1 ab.for

```fortran
      PROGRAM AB

      IMPLICIT REAL*8 (a-h,l-z)
      CHARACTER chA*2,chB*2,name*4,TorP*1
      INTEGER istA,istB,seein,seemid,maxout,densmol,howmany
      REAL*8
     &press1,press2,temp,temp1,temp2,delt,
     &MWA,DA2,IA2,TA2(4),ionA(5),guA(5,200,2),
     &MWB,DB2,IB2,TB2(4),ionB(5),guB(5,200,2),
     &Z0A(7,3),ZA(7,3),ubarA(7),uubA(7),rhsA(7),y0A,yA(7),
     &Z0B(7,3),ZB(7,3),ubarB(7),uubB(7),rhsB(7),y0B,yB(7),
     &enthA(7),entrA(7),CpA(7),
     &enthB(7),entrB(7),CpB(7),
     &RAB,IAB,TAB(4),
     &Z0AB(2,3),ZAB(2,3),ubarAB(2),uubAB(2),rhsAB(2),yAB(2),
     &enthAB(7),entrAB(7),CpAB(7),enthe,Cpe,entre,ye,yh,beta,
     &enthal,heatcp,entrop,mfA(7),mfB(7),mfAB(2),
     &denA(7),denB(7),denAB(2),dirA(7),dirB(7),dirAB(2),
     &mdot,Pabs,Pjet,thrust,Isp
      DATA tmp0 /2.9815E2/
      INCLUDE format.for
901   FORMAT (5x,'in.loop',3x,'wyhX-y0X',3x,'mid.loop',6x,'wyh-1',
     &3x,'out.loop',3x,'residual')
c
      WRITE(*,*)'Program: AB.FOR, ver. 1.3; Author: '//
     &'R.Nachtrieb, Jan 93;'
      WRITE(*,*)'OLAC-PL/RKFE Edwards AFB CA 93523-5000'
      WRITE(*,*)'internet email: nachtrieb@uiuc.edu'
      WRITE(*,*)
      WRITE(*,*)
c-----startup function gets temperatures, pressures, spectroscopic
c-----data from user and files
90    CALL startup(
     &press1,press2,TorP,temp1,temp2,howmany,
     &chA,istA,MWA,DA2,IA2,TA2,guA,ionA,y0A,
     &chB,istB,MWB,DB2,IB2,TB2,guB,ionB,y0B,
     &name,RAB,IAB,TAB,seein,seemid,maxout,mdot,densmol)
c-----get partition functions at STP, for later use in entropy difference
      CALL Z1(tmp0,istA,guA,TA2,ionA,Z0A,ubarA,uubA)
      CALL Z1(tmp0,istB,guB,TB2,ionB,Z0B,ubarB,uubB)
      CALL Z2(tmp0,Z0A,ubarA,uubA,Z0B,ubarB,uubb,TAB,Z0AB,ubarAB,uubAB)
```

```fortran
c-----either search for P(T)=Pwant or increment temperture
      IF (TorP.EQ.'P') THEN
c-------define temperature bounds
        Tmin=6e3
        Tmax=30e3
        Tlo=Tmin
        Thi=Tmax
        resP=1e7
        epsP=1e-2
c-------define lower absorbed power bound
        WRITE(*,*)'Getting lower absorbed-power bound...'
        WRITE(*,*)
        temp=Tmin
        CALL saha(
     &  press1,temp,istA,istB,seein,seemid,maxout,
     &  MWA,DA2,IA2,TA2,ionA,guA,
     &  MWB,DB2,IB2,TB2,ionB,guB,
     &  ZOA,ZA,ubarA,uubA,rhsA,y0A,yA,
     &  ZOB,ZB,ubarB,uubB,rhsB,y0B,yB,
     &  enthA,entrA,CpA,
     &  enthB,entrB,CpB,
     &  RAB,IAB,TAB,ZOAB,ZAB,ubarAB,uubAB,rhsAB,yAB,
     &  enthAB,entrAB,CpAB,enthe,Cpe,entre,ye,yh,beta)
        CALL gas(
     &  press1,temp,
     &  DA2,IA2,ionA,yA,enthA,CpA,entrA,
     &  DB2,IB2,ionB,yB,enthB,CpB,entrB,
     &  name,RAB,IAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &  enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &  dirA,dirB,dirAB)
        Pabs=enthal*mdot*1e3/(y0A*MWA+y0B*MWB)   !W
        Pmin=Pabs
c-------define lower absorbed power bound
        WRITE(*,*)'Getting upper absorbed-power bound...'
        WRITE(*,*)
        temp=Tmax
        CALL saha(
     &  press1,temp,istA,istB,seein,seemid,maxout,
     &  MWA,DA2,IA2,TA2,ionA,guA,
     &  MWB,DB2,IB2,TB2,ionB,guB,
     &  ZOA,ZA,ubarA,uubA,rhsA,y0A,yA,
     &  ZOB,ZB,ubarB,uubB,rhsB,y0B,yB,
```

```fortran
     &  enthA,entrA,CpA,
     &  enthB,entrB,CpB,
     &  RAB,IAB,TAB,ZOAB,ZAB,ubarAB,uubAB,rhsAB,yAB,
     &  enthAB,entrAB,CpAB,enthe,Cpe,entre,ye,yh,beta)
        CALL gas(
     &  press1,temp,
     &  DA2,IA2,ionA,yA,enthA,CpA,entrA,
     &  DB2,IB2,ionB,yB,enthB,CpB,entrB,
     &  name,RAB,IAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &  enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &  dirA,dirB,dirAB)
        Pabs=enthal*mdot*1e3/(y0A*MWA+y0B*MWB)   !W
        Pmax=Pabs
        WRITE(*,*)
        WRITE(*,'(A,1P2(E11.3))')')' Pmin,Pmax (W)',Pmin,Pmax
        WRITE(*,*)'What power to absorb? (W) '
        READ(*,*)Pwant
c-------check if requested power within power bounds
        IF (Pwant.LT.Pmin) THEN
          WRITE(*,*)'Sorry, requested power too low'
          WRITE(*,*)'Try decreasing the mass flow rate'
          WRITE(*,*)
          GOTO 90
        ELSEIF (Pwant.GT.Pmax) THEN
          WRITE(*,*)'Sorry, requested power too high'
          WRITE(*,*)'Try increasing the mass flow rate'
          WRITE(*,*)
          GOTO 90
        ENDIF
c-------perform temperature search
        WRITE(*,*)'  resP        epsP        temp',
     &  '                 out.loop    residual'
        WRITE(*,*)
        DO WHILE (resP.GT.epsP)
          IF (Pabs.LT.Pwant) THEN
          Tlo=temp
          temp=0.5*(temp+Thi)
          ELSEIF (Pwant.LT.Pabs) THEN
            Thi=temp
            temp=0.5*(Tlo+temp)
          ENDIF
          CALL saha(
```

```fortran
     &     press1,temp,istA,istB,seein,seemid,maxout,
     &     MWA,DA2,IA2,TA2,ionA,guA,
     &     MWB,DB2,IB2,TB2,ionB,guB,
     &     Z0A,ZA,ubarA,uubA,rhsA,y0A,yA,
     &     Z0B,ZB,ubarB,uubB,rhsB,y0B,yB,
     &     enthA,entrA,CpA,
     &     enthB,entrB,CpB,
     &     RAB,IAB,TAB,Z0AB,ZAB,ubarAB,uubAB,rhsAB,yAB,
     &     enthAB,entrAB,CpAB,enthe,Cpe,entre,ye,yh,beta)
           CALL gas(
     &     press1,temp,
     &     DA2,IA2,ionA,yA,enthA,CpA,entrA,
     &     DB2,IB2,ionB,yB,enthB,CpB,entrB,
     &     name,RAB,IAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &     enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &     dirA,dirB,dirAB)
           Pabs=enthal*mdot*1e3/(y0A*MWA+y0B*MWB)   !W
           resP=DABS((Pabs-Pwant)/Pwant)
993        FORMAT('+',1P3(E11.3))
           WRITE(*,993)resP,epsP,temp
           WRITE(*,*)
        ENDDO
        WRITE(*,*)
        WRITE(*,'(A,1P3(E11.3))')') Pwant,Pabs,temp',Pwant,Pabs,temp
        howmany=1
        itemp=1
c-------calculate ideal rocket performance
        CALL rocket(
     &   y0A,y0B,MWA,MWB,ye,yh,temp,press1,press2,mdot,
     &   thrust,Isp,Pjet)
        CALL output(
     &   y0A,y0B,temp1,temp2,howmany,press1,press2,mdot,temp,itemp,
     &   chA,ZA,rhsA,yA,enthA,CpA,entrA,
     &   chB,ZB,rhsB,yB,enthB,CpB,entrB,
     &   name,ZAB,rhsAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &   enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &   dirA,dirB,dirAB,Pabs,Pjet,thrust,Isp,densmol)
        WRITE(*,*)
      ELSEIF (TorP.EQ.'T') THEN
        delt=0.
        IF (howmany.NE.1) delt=(temp2-temp1)/DFLOAT(howmany-1)
        temp=temp1
```

```
          DO 100 itemp=1,howmany
            WRITE(*,192)' temp(K), press1(Pa)',temp,press1
            WRITE(*,901)
            WRITE(*,*)
c---------get equilibrium molecule/atom/ion fractions using partition function
          CALL saha(
     &      press1,temp,istA,istB,seein,seemid,maxout,
     &      MWA,DA2,IA2,TA2,ionA,guA,
     &      MWB,DB2,IB2,TB2,ionB,guB,
     &      ZOA,ZA,ubarA,uubA,rhsA,yOA,yA,
     &      ZOB,ZB,ubarB,uubB,rhsB,yOB,yB,
     &      enthA,entrA,CpA,
     &      enthB,entrB,CpB,
     &      RAB,IAB,TAB,ZOAB,ZAB,ubarAB,uubAB,rhsAB,yAB,
     &      enthAB,entrAB,CpAB,enthe,Cpe,entre,ye,yh,beta)
c---------calculate ideal rocket performance
          CALL rocket(
     &      yOA,yOB,MWA,MWB,ye,yh,temp,press1,press2,mdot,
     &      thrust,Isp,Pjet)
c---------get thermodynamic properties of the gas
          CALL gas(
     &      press1,temp,
     &      DA2,IA2,ionA,yA,enthA,CpA,entrA,
     &      DB2,IB2,ionB,yB,enthB,CpB,entrB,
     &      name,RAB,IAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &      enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &      dirA,dirB,dirAB)
c---------calculate maximum absorbed power
          Pabs=enthal*mdot*1e3/(yOA*MWA+yOB*MWB)   !W
c---------(J/mol)(kg/s)(g/kg)/(g/mol)=W
c---------write all accumulated data to file for later digestion
          CALL output(
     &      yOA,yOB,temp1,temp2,howmany,press1,press2,mdot,temp,itemp,
     &      chA,ZA,rhsA,yA,enthA,CpA,entrA,
     &      chB,ZB,rhsB,yB,enthB,CpB,entrB,
     &      name,ZAB,rhsAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &      enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &      dirA,dirB,dirAB,Pabs,Pjet,thrust,Isp,densmol)
          WRITE(*,*)
          temp=temp+delt
100     CONTINUE
c-------close all open files, tell user which files to examine for output
```

```
        ENDIF
        CALL shutdown(howmany,chA,chB,name,densmol)
        END   !main AB
C*******************************************************************
        INCLUDE STARTUP.FOR
        INCLUDE GETX.FOR
        INCLUDE GETXY.FOR
        INCLUDE NEWX.FOR
        INCLUDE NEWXY.FOR
        INCLUDE FILENAME.FOR
        INCLUDE OPUN.FOR
        INCLUDE CLOZE.FOR
        INCLUDE GETGU.FOR
        INCLUDE Z1.FOR
        INCLUDE Z2.FOR
        INCLUDE SAHA.FOR
        INCLUDE RHS1.FOR
        INCLUDE RHS2.FOR
        INCLUDE THERM1.FOR
        INCLUDE THERM2.FOR
        INCLUDE THERME.FOR
        INCLUDE GES1.FOR
        INCLUDE ITERY.FOR
        INCLUDE GETYE.FOR
        INCLUDE GETYH.FOR
        INCLUDE BALANCE.FOR
        INCLUDE NUY.FOR
        INCLUDE GETWYH.FOR
        INCLUDE RES1.FOR
        INCLUDE RES2.FOR
        INCLUDE ROCKET.FOR
        INCLUDE OUTPUT.FOR
        INCLUDE GAS.FOR
        INCLUDE SPECSHOW.FOR
        INCLUDE Z1SHOW.FOR
        INCLUDE Z2SHOW.FOR
        INCLUDE OPUNHDR.FOR
        INCLUDE SHUTDOWN.FOR
```

A.2   startup.for

```
C************************************************************************
      SUBROUTINE startup(
     &press1,press2,TorP,temp1,temp2,howmany,
     &chA,istA,MWA,DA2,IA2,TA2,guA,ionA,y0A,
     &chB,istB,MWB,DB2,IB2,TB2,guB,ionB,y0B,
     &name,RAB,IAB,TAB,seein,seemid,maxout,mdot,densmol)
      CHARACTER chA*2,chB*2,chdum*2,name*4,TorP*1
      INTEGER
     &istA,
     &istB,
     &seein,seemid,maxout,densmol,howmany
      REAL*8 press1,press2,temp1,temp2,
     &MWA,DA2,IA2,TA2(4),guA(5,200,2),ionA(5),y0A,
     &MWB,DB2,IB2,TB2(4),guB(5,200,2),ionB(5),y0B,
     &RAB,IAB,TAB(4)
      CHARACTER str*30
      IMPLICIT REAL*8 (a-h,l-z)
      INCLUDE format.for
      DATA atm /0.101325e6/
c-----Read in default inputs (stored from last run; if no file AB.in exists
c-----i.e., first time the program is run, program goes to label 100 and
c-----starts with the ''hard-wired'' defaults.
c-----User can select default by pressing <Enter>, or enter his own selection.
c-----After entering data, choices are written to file AB.in for next time.
      OPEN (UNIT=1, FILE='AB.IN')
      READ(1,*,ERR=100,END=100)chA,chB
      READ(1,*,ERR=100,END=100)y0A,y0B
      READ(1,*,ERR=100,END=100)press1,press2
      READ(1,*,ERR=100,END=100)TorP
      READ(1,*,ERR=100,END=100)temp1,temp2,howmany
      READ(1,*,ERR=100,END=100)densmol
      READ(1,*,ERR=100,END=100)seein,seemid,maxout
      READ(1,*,ERR=100,END=100)mdot
      GOTO 200
100   CONTINUE
      chA='C'
      chB='O'
      y0A=1.
      y0B=2.
      press1=1.
      press2=0.
```

```fortran
      TorP='T'
      temp1=6e3
      temp2=30e3
      howmany=1
      densmol=1
      seein=20
      seemid=20
      maxout=20
      mdot=1.
200   CONTINUE
c-----get element choices; case sensitive (thinks lower case is a new
c-----element, and prompts user for element data
300   WRITE(*,*)'Ar, C, H, N, O'
      WRITE(*,*)'Which two (2) elements to use?'
      WRITE(*,'('' ['',A,'','',''A,''] '')')chA,chB
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)chA,chB
      IF (chA.GT.chB) THEN
        chdum=chA
        chA=chB
        chB=chdum
      ENDIF
      OPEN (UNIT=10, FILE='ABX.in',STATUS='old')
c-----look in file ABX.in for information about generic unknown element
c-----''X'', where X is letter user typed.  If cannot find ''X'', in file
c-----ABX.in, get bomb!=0, and user has to either provide information
c-----about element X, or select new elements (start over).
c-----When data is satisfactorily known about X, data is assigned to (now)
c-----known element ''A'', which the rest of the program uses.  Process
c-----repeats for unknown X --> known B
      CALL getX(chA,istA,MWA,DA2,IA2,TA2,bomb)
      IF (bomb.NE.0.) THEN
        iok=1
        WRITE(*,*)
        WRITE(*,*)'Oops! Couldn''t find ',chA,' in file (ABX.in)'
        WRITE(*,*)'Make a new element? (1=yes)'
        WRITE(*,'('' ['',I1,'']'')')iok
        READ(*,'(A)')str
        IF (str.NE.' ') READ(str,*)iok
        IF (iok.EQ.1) THEN
c---------get the info about unknown element ''X''.  When finished, user
c---------has option of appending new element data to file ABX.in, so
```

```fortran
c---------element is ''known'' next time program is run.
          CALL newX(chA,istA,MWA,DA2,IA2,TA2)
        ELSE
          REWIND(UNIT=10)
          GOTO 300
        ENDIF
      ENDIF
      REWIND(UNIT=10)
      CALL getX(chB,istB,MWB,DB2,IB2,TB2,bomb)
      IF (bomb.NE.0.) THEN
        iok=1
        WRITE(*,*)
        WRITE(*,*)'Oops! Couldn''t find ',chB,' in file (ABX.in)'
        WRITE(*,*)'Make a new element? (1=yes)'
        WRITE(*,'('' ['',I1,'']'')')iok
        READ(*,'(A)')str
        IF (str.NE.' ') READ(str,*)iok
        IF (iok.EQ.1) THEN
          CALL newX(chB,istB,MWB,DB2,IB2,TB2)
        ELSE
          GOTO 300
        ENDIF
      ENDIF
      CLOSE(UNIT=10)
      IF ((chA(2:2).EQ.' ').AND.(chB(2:2).EQ.' ')) THEN
        name=chA(1:1)//chB(1:1)
      ELSEIF (chA(2:2).EQ.' ') THEN
        name=chA(1:1)//chB
      ELSEIF (chB(2:2).EQ.' ') THEN
        name=chA//chB(1:1)
      ELSE
        name=chA//chB
      ENDIF
      OPEN (UNIT=10, FILE='ABXY.in',STATUS='old')
      CALL getXY(name,RAB,IAB,TAB,bomb)
      IF (bomb.NE.0.) THEN
        iok=1
        WRITE(*,*)
        WRITE(*,*)'Oops! Couldn''t find ',name,' in file (ABXY.in)'
        WRITE(*,*)'Make a new compound? (1=yes)'
        WRITE(*,'('' ['',I1,'']'')')iok
        READ(*,'(A)')str
```

46

```fortran
      IF (str.NE.' ') READ(str,*)iok
      IF (iok.EQ.1) THEN
        CALL newXY(name,RAB,IAB,TAB)
      ELSE
        GOTO 300
      ENDIF
      ENDIF
      CLOSE(UNIT=10)
      WRITE(*,'(A)')'+Using '//chA//','//chB//','//name
      WRITE(*,*)
c-----get ratios of element A:B (parts of A vs. parts of B).  Converts
c-----to fractions of nuclei.
      WRITE(*,*)'Stoichiometric Proportions of '//chA//','//chB//'?'
      WRITE(*,'('' ['',1PE11.3,'','',1PE11.3,''] '')')y0A,y0B
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)y0A,y0B
      WRITE(*,192)'+Proportions of '//chA//','//chB,y0A,y0B
      IF((chA.EQ.chB).AND.(y0B.EQ.0.)) THEN
        y0A=0.
        y0B=1.
      ENDIF
      WRITE(*,*)
c-----Chamber pressure used for Saha eqs.  Exit pressure used for
c-----ideal rocket calculations.
      WRITE(*,*)'Chamber, exit pressures (atm)?'
      WRITE(*,'('' ['',1PE11.3,'','',1PE11.3,''] '')')press1,press2
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)press1,press2
      WRITE(*,192)'+press1,press2 (atm)',press1,press2
      WRITE(*,*)
c-----User specifies temperature (calculates absorbed power) or
c-----users specifies absorbed power (searches for corresponding temp)
500   WRITE(*,*)'Specify Temperature (T) or Absorbed Power (P)? '
      WRITE(*,'('' ['',A,''] '')')TorP
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)TorP
      IF ((TorP.EQ.'p').OR.(TorP.EQ.'P')) THEN
        TorP='P'
      ELSEIF ((TorP.EQ.'t').OR.(TorP.EQ.'T')) THEN
        TorP='T'
      ELSE
        WRITE(*,*)'What the... Try again!'
```

```fortran
        GOTO 500
      ENDIF
501   FORMAT('+',A)
      WRITE(*,501)TorP
      IF (TorP.EQ.'T') THEN
c-------get chamber temperature range (independent variable iterated)
c-------If howmany=1, program operates on first tempature only, regardless
c-------of upper temperature.
        WRITE(*,*)'Chamber temperature range: ',
     &   'temp1(K),temp2(K),howmany?'
        WRITE(*,'('' ['',1PE11.3,'','',1PE11.3,'','',I5,''] '')')
     &   temp1,temp2,howmany
        READ(*,'(A)')str
        IF (str.NE.' ') READ(str,*)temp1,temp2,howmany
        WRITE(*,'(A,1P2(1PE11.3),I5)')
     &   '+temp1(K),temp2(K),howmany',temp1,temp2,howmany
      ENDIF
c-----all interesting information is printed to file if running only
c-----one temperature; if multiple temperatures are run, program prints
c-----output data to separate files; due to file limitations of MS-DOS,
c-----user has to choose between molefraction and density outputs. Sigh...
      IF (howmany.GT.1) THEN
        WRITE(*,*)
        WRITE(*,*)'Print density (1) or mole fraction (2) ?'
        WRITE(*,'('' ['',I1,''] '')')densmol
        READ(*,'(A)')str
        IF (str.NE.' ') READ(str,*)densmol
        WRITE(*,'(''+densmol ['',I1,''] '')')densmol
      ENDIF
      WRITE(*,*)
c-----On personal computers, saha equations may take a long time to converge,
c-----especially if no numeric co-processor is used.  To entertain the user,
c-----and reassure them that the program is running (instead of hanging),
c-----variables seein,seemid, and maxout count the number of times the inner,
c-----middle, and outer nested while loops proceed before pooping out to the
c-----screen the loop count and the convergance.  If the computer is fast,
c-----printing out to the screen every loop really slows the program down;
c-----but if the computer is slow, the user can't tell if the program is
c-----working or not.
c-----Sugesstions: fast machines, set seein,seemid,maxout=100,100,20
c-----slow machines, seein,seemid,maxout=20,20,20.
c-----The program is ''done'' when the final residual (sum of all the
```

```fortran
c-----absolute values of the saha equation lhs-rhs) is less than epsilon,
c-----where epsilon=1e-7.  When the outer loop residual gets down around
c-----1e-6, it starts bouncing around.  1e-6 is still pretty small, so if
c-----the program isn't done before maxout outer loops, the program stops.
      WRITE(*,*)'View loop progress: inner,middle,outer?'
      WRITE(*,'('' ['',I4,'','',I4,'','',I4,''] '')')
     &seein,seemid,maxout
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)seein,seemid,maxout
      WRITE(*,113)'+seein,seemid,maxout', seein,seemid,maxout
      WRITE(*,*)
c-----for rocket stuff
      WRITE(*,*)'mass flow rate (kg/s)?'
      WRITE(*,'('' ['',1PE11.3,''] '')')mdot
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)mdot
      WRITE(*,111)'+mdot:',mdot
      WRITE(*,*)
      REWIND(UNIT=1)
c-----write info to file AB.in for next time defaults
      WRITE(1,*)chA,' ',chB
      WRITE(1,*)y0A,y0B
      WRITE(i,*)press1,press2
      WRITE(1,*)TorP
      WRITE(1,*)temp1,temp2,howmany
      WRITE(1,*)densmol
      WRITE(1,*)seein,seemid,maxout
      WRITE(1,*)mdot
      CLOSE(UNIT=1)
c-----convert atm's to Pa's
      press1=press1*atm
      press2=press2*atm
      total=y0A+y0B
      y0A=y0A/total
      y0B=y0B/total
c-----open appropriate data files and get atomic spectroscopic data;
c-----load spectroscopic data into array name ''gu'': g for degeneracy,
c-----u for electronic excitation energy level; ionization energies,
c-----in cm^-1, go into array ''ion''
      CALL opun(chA,istA)
      CALL getgu(istA,chA,guA,ionA)
      CALL cloze
```

```
      CALL opun(chB,istB)
      CALL getgu(istB,chB,guB,ionB)
      CALL cloze
      END !subroutine startup
C*********************************************************************
```

A.3   getx.for

```
c--get data from file ABX.in (unit 10) for unknown element X:
c--using element name (ch), gets highest element state to consider, i.e.
c--highest ionization level + 1 (ist), gets molecular weight (MW, in
c--g/mol), gets disso^ation (uD) and ionization (uI) energies of the
c--diatomic,homonuclear molecule, in eV, and gets the characteristic
c--temperatures of rotation and vibration for the neutral and singly
c--ionized diatomic molecule (in K).  If cannot find ch in the file,
c--sets bomb to non-zero, which flags the program outside this routine.
C******************************************************************
      SUBROUTINE getX(
     &ch,ist,MW,uD,uI,charT,bomb)
      CHARACTER*2 ch,readch
      INTEGER ist
      REAL*8 MW,uD,uI,charT(4),bomb
      bomb=0.
      READ(10,*)
      READ(10,*,ERR=100,END=100)readch,ist,MW,uD,uI,charT
      DO WHILE (readch.NE.ch)
         READ(10,*,ERR=100,END=100)readch,ist,MW,uD,uI,charT
      ENDDO
      RETURN
100   bomb=1.
      END
C******************************************************************
```

A.4   getxy.for

```
c--using the diatomic molecule name, composed of the compressed,
c--concatenated element names, in alphabetical order, subroutine looks
c--in file ABXY.in for information about the molecule.  Gets: reaction
c--(RAB) and ionization (IAB) energy of diatomic, heteronuclear
c--molecule, and the  characteristic temperatures for rotation and
c--vibration for the neutral and singly ionized molecule (both at ground
c--electronic state).  If cannot find info, bomb!=1.  If molecule is
c--pseudo-heteronuclear, e.g. it looks for diatomic nitrogen, even
c--though N2 is homonuclear, homonuclear info is returned.
C***************************************************************************
      SUBROUTINE getXY(
     &name,RAB,IAB,TAB,bomb)
      CHARACTER*4 name,readname
      REAL*8 RAB,IAB,TAB(4),bomb
      bomb=0.
      READ(10,*)
      READ(10,*,ERR=100,END=100)readname,RAB,IAB,TAB
      DO WHILE (readname.NE.name)
        READ(10,*,ERR=100,END=100)readname,RAB,IAB,TAB
      ENDDO
      RETURN
100   bomb=1.
      END
C***************************************************************************
```

A.5 newx.for

```
c--Prompts user for information about new elements.  Asks for
c--information that would otherwise be found in file ABX.in.  After
c--completion, gives user opportunity to append new element info to file
c--ABX.in, so next time the information will be found.  Gets highest
c--ionization state (ist), molecular weight (MW), dissocation (uD) and
c--ionization (uI) energies of diatomic homonuclear molecule, in eV, and
c--characteristic temperatures of rotation and vibration, at the ground
c-electronic state, of the neutral and singly ionized diatomic molecule.
C********************************************************************
      SUBROUTINE newX(
     &ch,ist,MW,uD,uI,charT)
      CHARACTER ch*2,str*30
      INTEGER ist
      REAL*8 MW,uD,uI,charT(4),iok
10    WRITE(*,*)'Let''s make a new element...'
      ist=0
      MW=1.
      uD=-50.
      uI=50.
      charT(1)=1.
      charT(2)=1.
      charT(3)=1.
      charT(4)=1.
      WRITE(*,*)
      WRITE(*,*)'Highest atomic state (1+highest ionization state)? '
      WRITE(*,'('' ['',I1,'']'')')ist
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)ist
      WRITE(*,'(''+ist: ['',I1,'']'')')ist
      WRITE(*,*)
      WRITE(*,*)'Atomic mass (amu)? '
      WRITE(*,'('' ['',1PE11.3,'']'')')MW
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)MW
      WRITE(*,'(''+MW: ['',1PE11.3,'']'')')MW
      WRITE(*,*)
      WRITE(*,*)'Dissociation energy of diatomic, homonuclear '//
     &'molecule (eV) ?'
      WRITE(*,'('' ['',1PE11.3,'']'')')uD
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)uD
```

```
 WRITE(*,'(''+uD: ['',1PE11.3,'']'')')uD
 WRITE(*,*)
 WRITE(*,*)'Ionization energy of diatomic, homonuclear '//
&'molecule (eV) ?'
 WRITE(*,'('' ['',1PE11.3,'']'')')uI
 READ(*,'(A)')str
 IF (str.NE.' ') READ(str,*)uI
 WRITE(*,'(''+uI: ['',1PE11.3,'']'')')uI
 WRITE(*,*)
 WRITE(*,*)'Characteristic temperatures, at ground electronic'//
&'state (K) ?'
 WRITE(*,*)'Neutral: Rotation, Vibration; Singly Ionized: '//
&'Rotation, Vibration'
 WRITE(*,'('' ['',1P4(E11.3),'']'')')charT
 READ(*,'(A)')str
 IF (str.NE.' ') READ(str,*)charT
 WRITE(*,'(''+charT: ['',1P4(E11.3),'']'')')charT
 WRITE(*,*)
 WRITE(*,*)'Summary:'
 WRITE(*,*)'Highest atomic state (1+highest ionization state) '
 WRITE(*,'('' ['',I1,'']'')')ist
 WRITE(*,*)'Atomic mass (amu) '
 WRITE(*,'('' ['',1PE11.3,'']'')')MW
 WRITE(*,*)'Dissociation energy of diatomic, homonuclear '//
&'molecule (eV) '
 WRITE(*,'('' ['',1PE11.3,'']'')')uD
 WRITE(*,*)'Ionization energy of diatomic, homonuclear '//
&'molecule (eV) '
 WRITE(*,'('' ['',1PE11.3,'']'')')uI
 WRITE(*,*)'Characteristic temperatures, at ground electronic'//
&'state (K) '
 WRITE(*,*)'Neutral: Rotation, Vibration; Singly Ionized: '//
&'Rotation, Vibration'
 WRITE(*,'('' ['',1P4(E11.3),'']'')')charT
 iok=1
 WRITE(*,*)'Is everything ok? (1=ok,other=redo)'
 WRITE(*,'('' ['',I1,'']'')')iok
 READ(*,'(A)')str
 IF (str.NE.' ') READ(str,*)iok
 IF (iok.NE.1) GOTO 10
 iok=0
 WRITE(*,*)'Good.  Append data to file (ABX.in)? (0=no,1=yes)'
```

```fortran
        WRITE(*,'('' ['',I1,'']'')')iok
        READ(*,'(A)')str
        IF (str.NE.' ') READ(str,*)iok
        IF (iok.EQ.1) THEN
          WRITE(*,*)'Appending data to file (ABX.in)...'
          WRITE(10,'(A,I2,1P7(E11.3))') ch,ist,MW,uD,uI,charT
        ELSE
          WRITE(*,*)'Will not append data to file...'
        ENDIF
        END !subroutine newX
C******************************************************************************
```

A.6 newxy.for

```
c--Asks user for information about unknown compound.  Gets reaction
c--(RAB) and ionization (IAB) energy of heteronuclear diatomic molecule,
c--in eV.  Also gets characteristic temperatures of rotation and
c--vibration, at the ground electronic state, of the diatomic neutral
c--and singly ionized molecule.  User can append data to end of input
c--file ABXY.in to save typing data in again next time.
C*******************************************************************
      SUBROUTINE newXY(
     &name,RAB,IAB,TAB)
      CHARACTER name*4,str*30
      REAL*8 RAB,IAB,TAB(4)
      WRITE(*,*)
10    WRITE(*,*)'Let''s make a new compound...'
      RAB=-50.
      IAB=50.
      TAB(1)=1.
      TAB(2)=1.
      TAB(3)=1.
      TAB(4)=1.
      WRITE(*,*)
      WRITE(*,*)'Reaction energy of diatomic, heteronuclear '//
     &'molecule (eV) ?'
      WRITE(*,'('' ['',1PE11.3,'']'')')RAB
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)RAB
      WRITE(*,'(''+RAB: ['',1PE11.3,'']'')')RAB
      WRITE(*,*)
      WRITE(*,*)'Ionization energy of diatomic, heteronuclear '//
     &'molecule (eV) ?'
      WRITE(*,'('' ['',1PE11.3,'']'')')IAB
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)IAB
      WRITE(*,'(''+IAB: ['',1PE11.3,'']'')')IAB
      WRITE(*,*)
      WRITE(*,*)'Characteristic temperatures, at ground electronic'//
     &'state (K) ?'
      WRITE(*,*)'Neutral: Rotation, Vibration; Singly Ionized: '//
     &'Rotation, Vibration'
      WRITE(*,'('' ['',1P4(E11.3),'']'')')TAB
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)TAB
```

```fortran
      WRITE(*,'(''+TAB: ['',1P4(E11.3),'']'')')TAB
      iok=1
      WRITE(*,*)
      WRITE(*,*)'Reaction energy of diatomic, heteronuclear '//
     &'molecule (eV) '
      WRITE(*,'('' ['',1PE11.3,'']'')')RAB
      WRITE(*,*)'Ionization energy of diatomic, heteronuclear '//
     &'molecule (eV) '
      WRITE(*,'('' ['',1PE11.3,'']'')')IAB
      WRITE(*,*)'Characteristic temperatures, at ground electronic'//
     &'state (K) '
      WRITE(*,*)'Neutral: Rotation, Vibration; Singly Ionized: '//
     &'Rotation, Vibration'
      WRITE(*,'('' ['',1P4(E11.3),'']'')')TAB
      WRITE(*,*)'Is everything ok? (1=ok,other=redo)'
      WRITE(*,'('' ['',I1,'']'')')iok
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)iok
      IF (iok.NE.1) GOTO 10
      iok=0
      WRITE(*,*)'Good.  Append data to file (ABXY.in)? (0=no,1=yes)'
      WRITE(*,'('' ['',I1,'']'')')iok
      READ(*,'(A)')str
      IF (str.NE.' ') READ(str,*)iok
      IF (iok.EQ.1) THEN
        WRITE(*,*)'Appending data to file (ABXY.in)...'
        WRITE(10,'(A,1P6(E11.3))')name,RAB,IAB,TAB
      ELSE
        WRITE(*,*)'Will not append data to file...'
      ENDIF
      END
C*******************************************************************
```

A.7 opun.for

```
c--open the necessary files to get the atomic spectra data.  The naming
c--convention is as follows: the input file names start with ch (removed
c--of spaces), have an underscore (''_''), and then an number, 0-4,
c--which corresponds to the charge on the atom.  Charge +4 corresponds
c--to ist=5, e.g. C_V, carbon ionized four times.
C**********************************************************************
      SUBROUTINE opun(
     &ch,ist)
      CHARACTER ch*2,filenum*1,filename*12
      INTEGER i
      DO 10 i=0,ist-1
        WRITE(filenum,'(I1)')i
        OPEN(UNIT=10+i,
     &  FILE=filename(ch,'_'//filenum//'.in'),
     &  STATUS='OLD')
10    CONTINUE
      END !subroutine opun
C**********************************************************************
```

A.8   cloze.for

```
c--close all the opened files
C*********************************************************************
      SUBROUTINE cloze
      CLOSE(UNIT=10)
      CLOSE(UNIT=11)
      CLOSE(UNIT=12)
      CLOSE(UNIT=13)
      CLOSE(UNIT=14)
      END !subroutine opun
C*********************************************************************
```

A.9 getgu.for

```
c--necessary files are already opened, so reads from files unit=10
c--through unit=10+ist-1 to get the ionization energy of the atom, in
c--cm^-1, and the (degeneracy, energy level) pairs (energy levels also
c--in cm^-1).  Ionization energies stored in array ''ion'', and
c--degeneracy, level pairs stored in array ''gu''.  For example,
c--guA(5,200,2) stores the 200x2 (g,u) pairs for the 5 species of
c--element A.
C*****************************************************************
      SUBROUTINE getgu(
     &ist,ch,gu,ion)
       INTEGER ist
       CHARACTER ch*2,filenum*1,filename*12
       REAL*8 gu(5,200,2),ion(5)
       INTEGER i,j,k
       IMPLICIT REAL*8 (a-h,l-z)
       DO 100 i=1,5
         DO 100 j=1,200
           DO 100 k=1,2
100            gu(i,j,k)=0.
       DO 200 i=1,ist
         WRITE(filenum,'(I1)')i-1
         WRITE(*,*)'Getting data from file '//
     &   filename(ch,'_'//filenum//'.in')
         idisk=9+i
         READ(idisk,*) ion(i)
         j=0
         READ(idisk,*) degen,englev
         DO WHILE (degen.GT.0.)
            j=j+1
            gu(i,j,1)=degen
            gu(i,j,2)=englev
            READ(idisk,*) degen,englev
         ENDDO
200    CONTINUE
       END  !subroutine getgu
C*****************************************************************
```

A.10 saha.for

```
c--The meat of the program.  Subroutine has 3 nested while loops, one
c--of which is hidden in SUBROUTINE balance.  Outer loop updates
c--electron and heavy fractional densities (ye, yh), and beta
c--(beta=ye/(yh+ye)).  Middle loop holds ye,yh,beta constant, but
c--updates heteronuclear reaction products AB, AB+.  Inner loops update
c--atomic and homonuclear diatomic species.  (See SUBROUTINE balance for
c--inner loops.)
C*******************************************************************
      SUBROUTINE saha(
     &press1,temp,istA,istB,seein,seemid,maxout,
     &MWA,DA2,IA2,TA2,ionA,guA,
     &MWB,DB2,IB2,TB2,ionB,guB,
     &ZOA,ZA,ubarA,uubA,rhsA,yOA,yA,
     &ZOB,ZB,ubarB,uubB,rhsB,yOB,yB,
     &enthA,entrA,CpA,
     &enthB,entrB,CpB,
     &RAB,IAB,TAB,ZOAB,ZAB,ubarAB,uubAB,rhsAB,yAB,
     &enthAB,entrAB,CpAB,enthe,Cpe,entre,ye,yh,beta)
      IMPLICIT REAL*8 (a-h,l-z)
      INTEGER istA,istB,seein,seemid,maxout
      REAL*8 press1,temp,
     &MWA,DA2,IA2,TA2(4),ionA(5),guA(5,200,2),
     &MWB,DB2,IB2,TB2(4),ionB(5),guB(5,200,2),
     &ZOA(7,3),ZA(7,3),ubarA(7),uubA(7),rhsA(7),yOA,yA(7),
     &ZOB(7,3),ZB(7,3),ubarB(7),uubB(7),rhsB(7),yOB,yB(7),
     &enthA(7),entrA(7),CpA(7),
     &enthB(7),entrB(7),CpB(7),
     &RAB,IAB,TAB(4),
     &ZOAB(2,3),ZAB(2,3),ubarAB(2),uubAB(2),rhsAB(2),yAB(2),
     &enthAB(7),entrAB(7),CpAB(7),
     &ye,yh,beta
C     local variables for subroutine saha
      INTEGER i,imid,iout,ityA,ityB
      REAL*8
     &epsilon,chi,ymin,
     &yeA,yhA,resA,
     &yeB,yhB,resB,
     &resAB,wyh,residual
      DATA epsilon,chi,ymin /1.E-7,0.5,1.E-26/
c-----get the partition functions for species of A,B, and AB for the
c-----given temperature and pressure.
```

```fortran
      CALL Z1(temp,istA,guA,TA2,ionA,ZA,ubarA,uubA)
      CALL Z1(temp,istB,guB,TB2,ionB,ZB,ubarB,uubB)
      CALL Z2(temp,ZA,ubarA,uubA,ZB,ubarB,uubb,TAB,ZAB,ubarAB,uubAB)
c-----get the right-hand-sides (rhs) of the saha equations (R_i in the
c-----report), which are specified for a given temperature and pressure.
      CALL rhs1(temp,press1,istA,ionA,ZA,DA2,IA2,MWA,rhsA)
      CALL rhs1(temp,press1,istB,ionB,ZB,DB2,IB2,MWB,rhsB)
      CALL rhs2(temp,press1,ZA,ZB,ZAB,RAB,IAB,MWA,MWB,rhsAB)
c-----get thermodynamic properties for the individual species; depend
c-----only on temp, press, partition functions, and ubar, uub; can get
c-----whole gas thermodynamic properties later by taking the weighted
c-----average thermodynamic properites, where the weights are the mole
c-----fractions.
      CALL therm1(temp,press1,DA2,IA2,ionA,ZA,Z0A,ubarA,uubA,
     &enthA,CpA,entrA)
      CALL therm1(temp,press1,DB2,IB2,ionB,ZB,Z0B,ubarB,uubB,
     &enthB,CpB,entrB)
      CALL therm2(temp,press1,DA2,DB2,RAB,IAB,ZAB,Z0AB,
     &ubarAB,uubAB,enthAB,CpAB,entrAB)
      CALL therme(temp,press1,enthe,Cpe,entre)
      DO 200 i=1,7
        yA(i)=0.
        yB(i)=0.
200   CONTINUE
      DO 300 i=1,2
        yAB(i)=0.
300   CONTINUE
c-----guess (''ges'') the initial values of the atomic and homonuclear
c-----atomic species fractions by assuming that (1) elements A,B can
c-----mix, but do not react chemically, (2) a given element has two
c-----dominant species present at any one time, all others being
c-----negligibly (sp?) small.  For a single element, or for mixtures,
c-----this method will give the right answer to about 5%, and zippy
c-----quick too!  But chemical reactions between A and B require less
c-----elegant methods...
      CALL ges1(rhsA,y0A,yA)
      CALL ges1(rhsB,y0B,yB)
c-----itery finds the dominant species in an element, and sets the
c-----variably ity to that species position in the element array.  In
c-----subsequent calculations, operations are perfoimed using that
c-----species as the ''row pivot''.  This reduces numerical error,
c-----which means the set of equations converges more quickly to a good
```

```
c-----answer.
      CALL itery(yA,ityA)
      CALL itery(yB,ityB)
c-----functions that get the sum of the atomic fractions for the
c-----''heavy'' fraction (yh), and the weighted average atomic
c-----fractions for the electron fraction (ye), where the weights are
c-----the species charges.  At this point in the program, this is just
c-----our first guess at ye,yh,and beta.
      CALL getye(yA,yeA)
      CALL getye(yB,yeB)
      CALL getyh(yA,yhA)
      CALL getyh(yB,yhB)
      ye=yeA+yeB+yAB(2)
      yh=yhA+yhB+yAB(1)+yAB(2)
      beta=ye/(yh+ye)
c-----calculate the maximum possible fraction of species AB, as if all
c-----of one species or another combined to form the compound; maximum
c-----is modified by the rhsAB, which sort of indicates how much the
c-----species AB dissociates as the temperature increases.
      ymaxAB=0.5-DSQRT(0.25-y0A*y0B/(1.+rhsAB(1)))
      iout=0
      residual=1.
c-----outer while loop: finished when either (1) residual<epsilon, which
c-----means the y's we have satisfies the system of equations nicely,
c-----or (2) the loop has tried maxout times to converge, and hasn't
c-----done it yet, and so probably won't.  maxout is definable by the
c-----user in the input section: maxout=20 works well, and reasonably
c-----balances speed vs. efficiency.  This loop balances charge.
      DO WHILE ((residual.GT.epsilon).AND.(iout.LT.maxout))
        ye0=ye
        yh0=yh
        imid=0
c-------reinitialize the upper and lower bounds for yAB
        yhiAB=ymaxAB
        yloAB=0.
        wyh=-1.
c-------middle while loop: is done when the weighted sum of the heavy
c-------atomic fractions (wyh) is within epsilon of 1, where the
c-------weights are the number of nuclei in the molecule (2 for
c-------diatomic, 1 for atomic).  This is essentially the conservation
c-------of mass equation.  Within this loop, charge is held temporarily
c-------constant.
```

```fortran
        DO WHILE (DABS(wyh-1.).GT.epsilon)
          yAB0=yAB(1)
c---------inner loop(s), one each for the elements A and B: is done when
c---------the weighted sum of the atomic fractions (sum of element
c---------nuclei) is within epsilon of the total atomic fraction, y0;
c---------This is essentially conservation of mass for each element.
          CALL balance(ye,yh,beta,epsilon,chi,ymin,yAB,seein,
     &    y0A,yA,ityA,rhsA)
          CALL balance(ye,yh,beta,epsilon,chi,ymin,yAB,seein,
     &    y0B,yB,ityB,rhsB)
c---------done with inner loops, so now update yAB, yAB+,
c---------under-relaxing using bisection method.  Converges quickly for
c---------monotonic functions.
          yAB(1)=yA(1)*yB(1)/(yh+ye)/rhsAB(1)
          yAB1=yAB(1)
          IF (yAB1.GT.yAB0) THEN
            yloAB=yAB0
            yAB(1)=(1.-chi)*yAB0+chi*yhiAB
          ELSEIF (yAB1.LT.yAB0) THEN
            yhiAB=yAB0
            yAB(1)=(1.-chi)*yAB0+chi*yloAB
          ENDIF
          IF (ye.EQ.0.) THEN
            yAB(2)=0
          ELSEIF (ye.GT.0.) THEN
            yAB(2)=yAB(1)*rhsAB(2)/beta
          ENDIF
c---------get new weighted sums of heavies for middle loop check
          CALL getwyh(yA,yAB,wyhA)
          CALL getwyh(yB,yAB,wyhB)
          wyh=wyhA+wyhB
          imid=imid+1
922       FORMAT ('+',22x,1P2(E11.3))
c---------the progress-o-meter
          IF (mod(imid,seemid).EQ.0) THEN
            WRITE(*,922)imid,wyh-1.
          ENDIF
        ENDDO
c-------get new electron, heavy fractions for the outer loop check
        CALL getye(yA,yeA)
        CALL getye(yB,yeB)
        CALL getyh(yA,yhA)
```

```fortran
      CALL getyh(yB,yhB)
      ye=yeA+yeB+yAB(2)
      yh=yhA+yhB+yAB(1)+yAB(2)
      ye1=ye
      yh1=yh
c-------update electron, heavy density fractions by under-relaxing,
c-------using bisection method.  Again, best for monotonic functions
c-------(such as we have).
      ye=(1.-chi)*ye0+chi*ye1
      yh=(1.-chi)*yh0+chi*yh1
      beta=ye/(yh+ye)
c-------get partial and total residuals for outer loop check
      CALL res1(y0A,yA,rhsA,yAB,ye,yh,beta,resA)
      CALL res1(y0B,yB,rhsB,yAB,ye,yh,beta,resB)
      CALL res2(yA,yB,yAB,rhsAB,ye,yh,beta,resAB)
      residual=resA+resB+resAB
      iout=iout+1
944   FORMAT ('+',44x,1P2(E11.3))
c-------are we almost there?
      WRITE(*,944)iout,residual
      ENDDO
      END !subroutine saha
C***************************************************************************
```

A.11   z1.for

```
c--Make the partition function for the element, given the atomic spectra
c--data in arrays gu and ion, and the temperature.  Also calculates the
c--average electronic excitation energy level, ubar, and the average
c--squared energy excitation energy level, uub (short for (u*u)bar).
c--These are useful later in calculateing thermodynamic properties, such
c--as heat capacity.  Array ZA(7,3), holds, for example, partition
c-function data for the 7 states (atomic neutral, atomic +1,...+4,
c--diatomic neutral, diatomic +1) of element A, for electronic,
c--rotational, and vibrational excitation (Zrot and Zvib =1 for atomic
c--species).  Electronic partition functions for diatomic species are
c--approximated by appropriately averaged atomic partition functions.
c--For example, Zel,A2+ \approx (ZelA)*(ZelA+).  Remember, partition
c--functions multiply, not add.
C*********************************************************************
      SUBROUTINE Z1(
     &temp,ist,gu,charT,ion,Z,ubar,uub)
      INTEGER ist
      REAL*8 temp,gu(5,200,2),charT(4),ion(5),Z(7,3),ubar(7),uub(7)
      INTEGER i,j
      IMPLICIT REAL*8 (a-h,l-z)
      tempinv=1.4387/temp
      DO 100 i=1,7
        DO 100 j=1,3
100       Z(i,j)=0
      DO 200 i=1,ist
        summ=0.
        sumu=0.
        sumuu=0.
        j=1
        DO WHILE (gu(i,j,1).GT.0.)
          term = 0.
          IF((gu(i,j,2)*tempinv.LT.70.)
     &    .AND.(gu(i,j,2).LT.ion(i)-temp/1.4387))
     &      term=gu(i,j,1)*DEXP(-gu(i,j,2)*tempinv)
          summ =summ +term
          sumu =sumu +term*gu(i,j,2)
          sumuu=sumuu+term*gu(i,j,2)*gu(i,j,2)
          j=j+1
        ENDDO
        Z(i,1)=summ
        Z(i,2)=1.
```

```fortran
          Z(i,3)=1.
          ubar(i)=sumu/summ
          uub(i)=sumuu/summ
200       CONTINUE
          Z(6,1)=Z(1,1)*Z(1,1)
          Z(6,2)=temp/charT(1)
          Z(6,3)=temp/charT(2)
          ubar(6)=ubar(1)+ubar(1)
          uub(6)=uub(1)-ubar(1)*ubar(1)+uub(1)-ubar(1)*ubar(1)
          Z(7,1)=Z(1,1)*Z(2,1)
          Z(7,2)=temp/charT(3)
          Z(7,3)=temp/charT(4)
          ubar(7)=ubar(1)+ubar(2)
          uub(7)=uub(1)-ubar(1)*ubar(1)+uub(2)-ubar(2)*ubar(2)
          END   !subroutine Z
C***************************************************************************
```

A.12 z2.for

```
c--Gets partition functions for diatomic species (hence the 2 in Z2).
c--Approximates the electronic partition function by appropriately
c--average atomic partition functions.  For example, Zel,AB+ is an
c--average between Zel,A, Zel,B+, Zel,A+, and Zel,B.  Although this is
c--not rigorously correct, it gets in the ballpark;  I don't worry too
c--much about diatomic species since (1) this code is meant to be run at
c--high temperatures, where there aren't many diatomic species, and (2)
c--Selph's Isp code does a great (and better, too) job at lower
c--temperatures.  Also gets averaged ubar and (u*u)bar for diatomic
c--species.  Note that of ZA=ZB, ZAB=ZA2=ZB2, as should be.  Thus,
c--entering in two identical species does not confuse the code.
C********************************************************************
      SUBROUTINE Z2(
     &temp,ZA,ubarA,uubA,ZB,ubarB,uubb,TAB,ZAB,ubarAB,uubAB)
      REAL*8 temp,
     &ZA(7,3),ubarA(7),uubA(7),
     &ZB(7,3),ubarB(7),uubB(7),
     &TAB(4),ZAB(2,3),ubarAB(2),uubAB(2)
      IMPLICIT REAL*8 (a-h,l-z)
      ZAB(1,1)=ZA(1,1)*ZB(1,1)
      ZAB(1,2)=temp/TAB(1)
      ZAB(1,3)=temp/TAB(2)
      ubarAB(1)=ubarA(1)+ubarB(1)
      uubAB(1)=uubA(1)-ubarA(1)*ubarA(1)+uubB(1)-ubarB(1)*ubarB(1)
      ZAB(2,1)=0.5*(ZB(2,1)*ZA(1,1)+ZA(2,1)*ZB(1,1))
      ZAB(2,2)=temp/TAB(3)
      ZAB(2,3)=temp/TAB(4)
      ubarAB(2)=0.5*(ubarA(1)+ubarA(2)+ubarB(1)+ubarB(2))
      uubAB(2)=0.5*(
     &uubA(1)-ubarA(1)*ubarA(1)+uubB(1)-ubarB(1)*ubarB(1)+
     &uubA(2)-ubarA(2)*ubarA(2)+uubB(2)-ubarB(2)*ubarB(2))
      END   .subroutine Z2
C********************************************************************
```

A.13   therm1.for

c--gets the thermodynamic functions (enthalpy, heat capacity, and
c--entropy difference from STP) for an element's species, and sticks
c--them in arrays enth(7), Cp(7), and entr(7). Uses the element's
c--partion functions at STP (Z0) and the current T,P (Z). See Theory
c--in paper for derivation of thermodynamic functions.

```
C****************************************************************
      SUBROUTINE therm1(
     &temp,press1,uD,uI,ion,Z,Z0,ubar,uub,enth,Cp,entr)
      REAL*8 temp,press1,uD,ion(5),Z(7,3),Z0(7,3),ubar(7),uub(7),
     &enth(7),Cp(7),entr(7)
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      DATA gasR,cminv,atm/8.31439,8.06549E3,0.101325e6/
      DATA tmp0 /2.9815E2/
      tempinv=1.4387/temp
      dision=0.5*uD*cminv
      DO 100 i=1,5
        IF (Z(i,1).EQ.0) THEN
          enth(i)=0.
          Cp(i)=0.
          entr(i)=0.
        ELSE
          enth(i)=gasR*temp*(2.5+(dision+ubar(i))*tempinv)
          Cp(i)=gasR*(2.5+(uub(i)-ubar(i)*ubar(i))*tempinv*tempinv)
          entr(i)=gasR*(2.5*DLOG(temp/tmp0)+ubar(i)*tempinv+
     &    DLOG(Z(i,1)/Z0(i,1))-DLOG(press1/atm))
          dision=dision+ion(i)
        ENDIF
100   CONTINUE
      enth(6)=gasR*temp*(4.5+ubar(6)*tempinv)
      Cp(6)=gasR*(4.5+(uub(6)-ubar(6)*ubar(6))*tempinv*tempinv)
      entr(6)=gasR*(4.5*DLOG(temp/tmp0)+ubar(6)*tempinv+
     &DLOG(Z(6,1)/Z0(6,1))-DLOG(press1/atm))
      enth(7)=gasR*temp*(4.5+(uI*cminv+ubar(7))*tempinv)
      Cp(7)=gasR*(4.5+(uub(7)-ubar(7)*ubar(7))*tempinv*tempinv)
      entr(7)=gasR*(4.5*DLOG(temp/tmp0)+ubar(7)*tempinv+
     &DLOG(Z(7,1)/Z0(7,1))-DLOG(press1/atm))
      END
C****************************************************************
```

A.14 therm2.for

```
c--calculates the thermodynamic functions (h,c_p,s-s0) for the
c--heteronuclear diatomic species.  Sticks them in arrays enthAB(2),
c--CpAB(2), and entrAB(2).
C*****************************************************************
      SUBROUTINE therm2(
     &temp,press1,DA2,DB2,RAB,IAB,ZAB,ZOAB,
     &ubarAB,uubAB,enthAB,CpAB,entrAB)
      REAL*8 temp,press1,DA2,DB2,RAB,ZAB(2,3),ZOAB(2,3),
     &ubarAB(2),uubAB(2),enthAB(2),CpAB(2),entrAB(2)
      IMPLICIT REAL*8 (a-h,l-z)
      DATA gasR,cminv,atm/8.31439,8.06549E3,0.101325e6/
      DATA tmp0 /2.9815E2/
      tempinv=1.4387/temp
      dision=(0.5*(DA2+DB2)-RAB)*cminv
      enthAB(1)=gasR*temp*(4.5+(dision+ubarAB(1))*tempinv)
      CpAB(1)=gasR*(4.5+(uubAB(1)-ubarAB(1)*ubarAB(1))*tempinv*tempinv)
      entrAB(1)=gasR*(4.5*DLOG(temp/tmp0)+ubarAB(1)*tempinv+
     &DLOG(ZAB(1,1)/ZOAB(1,1))-DLOG(press1/atm))
      dision=(0.5*(DA2+DB2)-RAB+IAB)*cminv
      enthAB(2)=gasR*temp*(4.5+(dision+ubarAB(2))*tempinv)
      CpAB(2)=gasR*(4.5+(uubAB(2)-ubarAB(2)*ubarAB(2))*tempinv*tempinv)
      entrAB(2)=gasR*(4.5*DLOG(temp/tmp0)+ubarAB(2)*tempinv+
     &DLOG(ZAB(2,1)/ZOAB(2,1))-DLOG(press1/atm))
      END
C*****************************************************************
```

A.15   therme.for

```
c--get thermodynamic functions (h,c_p,s-s0) for electrons.
C*******************************************************************
      SUBROUTINE therme(
     &temp,press1,enthe,Cpe,entre)
      REAL*8 temp,press1,enthe,Cpe,entre
      IMPLICIT REAL*8 (a-h,l-z)
      DATA gasR,atm/8.31439,0.101325e6/
      DATA tmp0 /2.9815E2/
      enthe=2.5*gasR*temp
      Cpe=2.5*gasR
      entre=gasR*(2.5*DLOG(temp/tmp0)-DLOG(press1/atm))
      END
C*******************************************************************
```

A.16   rhs1.for

```
c--gets the right-hand-sides (rhs) of the Saha equation for use in the
c--nuclear fraction (y) calculations.  See paper theory for derivations.
c--Say there are the neutral atomic species, 4 ionized atomic species,
c--and the two diatomic species: 7 species per element.  We only need 6
c--equations, and therefore 6 rhs's, to solve for these 7 species,
c--since the seventh equation is conservation of mass.  Therefore, for
c--the 7 species we have, rhs(5) will be left empty (0), and rhs(6) and
c--rhs(7) correspond to the diatomic species equations.  If we only go
c--up to A+3, then rhs(4) and rhs(5) will be empty.
C*******************************************************************
      SUBROUTINE rhs1(
     &temp,press1,ist,ion,Z,uD,uI,MW,rhs)
      INTEGER ist
      REAL*8 temp,press1,ion(5),Z(7,3),uD,uI,MW,rhs(7)
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      DATA coefD,coefI,K_eV /2.59466E3,3.33381E-2,11604.85/
      tempinv=1.4387/temp
      tempre=temp*temp*DSQRT(temp)/press1
      DO 100 i=1,7
100      rhs(i)=0.
      DO 200 i=1,ist-1
        boltz=0.
        IF (ion(i)*tempinv.LT.70.) boltz=DEXP(-ion(i)*tempinv)
        rhs(i)=coefI*Z(i+1,1)/Z(i,1)*boltz*tempre
200   CONTINUE
      mu=MW*MW/(MW+MW)
      boltz=0.
      IF (uD*K_eV/temp.LT.70.) boltz=DEXP(-uD*K_eV/temp)
      rhs(6)=coefD*mu*DSQRT(mu)*boltz*tempre*
     &Z(1,1)*Z(1,1)/
     &Z(6,1)/Z(6,2)/Z(6,3)
      boltz=0.
      IF (uI*K_eV/temp.LT.70.) boltz=DEXP(-uI*K_eV/temp)
      rhs(7)=coefI*boltz*tempre*
     &Z(7,1)*Z(7,2)*Z(7,3)/
     &Z(6,1)/Z(6,2)/Z(6,3)
      END  !subroutine rhs1
C*******************************************************************
```

A.17 rhs2.for

```
c--gets right-hand-sides of Saha equations for diatomic, heteronuclear
c--species. See paper theory for derivation.
C*******************************************************************
      SUBROUTINE rhs2(
     &temp,press1,ZA,ZB,ZAB,RAB,IAB,MWA,MWB,rhsAB)
       REAL*8 temp,press1,ZA(7,3),ZB(7,3),ZAB(2,3),RAB,IAB,
     &MWA,MWB,rhsAB(2)
       IMPLICIT REAL*8 (a-h,l-z)
       DATA coefD,coefI,K_eV /2.59466E3,3.33381E-2,11604.85/
       tempre=temp*temp*DSQRT(temp)/press1
       boltz=0.
       IF (RAB*K_eV/temp.LT.70.) boltz=DEXP(-RAB*K_eV/temp)
       muAB=MWA*MWB/(MWA+MWB)
       rhsAB(1)=coefD*muAB*DSQRT(muAB)*boltz*tempre*
     &ZA(1,1)*ZB(1,1)/
     &ZAB(1,1)/ZAB(1,2)/ZAB(1,3)
       boltz=0.
       IF (IAB*K_eV/temp.LT.70.) boltz=DEXP(-IAB*K_eV/temp)
       rhsAB(2)=coefI*boltz*tempre*
     &ZAB(2,1)*ZAB(2,2)*ZAB(2,3)/
     &ZAB(1,1)/ZAB(1,2)/ZAB(1,3)
       END  !subroutine rhs2
C*******************************************************************
```

A.18   ges1.for

```
c--guesses (hence ''ges'') the initial species distribution for single
c--(hence ''1'') element, based on the assumption that only two species
c--dominate at at T,P.  This assumption linearizes the otherwise very
c--nonlinear equations, and gives the right answer to within 5% for a
c--single element, or for an inert (no chemical reactions) mixture of
c--multiple elements.  Lickity split.
c--The conditions for checking which two species are dominant are
c--these:  say we're examining N_+1, N_+2, and N_+3 (abbrv. N1, N2, N3)
c--The switchover point from considering N1,N2 to considering N2,N3
c--comes when there are equal concentrations of N1 and N3.  Using this,
c--the fact than N1+N2+N3=N_heavy, and the fact that
c--N1+2N2+3N3=N_electrons all give simple relations for the crossover
c--points; what makes the crossover points unique for an element are
c--the values of the right-hand-sides of the Saha equations, which
c--depend on the molecular weight, partition functions, etc.
C*****************************************************************************
      SUBROUTINE ges1(
     &rhs,y0,y)
      REAL*8 rhs(7),y0,y(7)
      IMPLICIT REAL*8 (a-h,l-z)
      IF (rhs(6)*rhs(1).LT.1e-3) THEN
        y(1)=y0*DSQRT(rhs(6)/(4.+rhs(6)))
        y(6)=y0-y(1)
      ELSEIF (rhs(1)*rhs(2).LT.0.25) THEN
        y(2)=y0/DSQRT(1+rhs(1))
        y(1)=y0-y(2)
      ELSEIF (rhs(2)*rhs(3).LT.0.44) THEN
        y(3)=y0*(-0.5+DSQRT(0.25+2.*rhs(2)/(1.+rhs(2))))
        y(2)=y0-y(3)
      ELSEIF (rhs(2)*rhs(3).GE.0.44) THEN
        y(4)=y0*(-1.+DSQRT(1.+3.*rhs(3)/(1.+rhs(3))))
        y(3)=y0-y(4)
      ENDIF
      CALL getye(y,ye)
      IF (ye.EQ.0.) THEN
        y(2)=y(1)*rhs(1)+DSQRT(y(1)*y(1)*rhs(1)*rhs(1)+
     &  y(1)*rhs(1)*(y(1)+y(6)))
        y(7)=y(6)*rhs(7)+DSQRT(y(6)*y(6)*rhs(7)*rhs(7)+
     &  y(6)*rhs(7)*(y(6)+y(1)))
      ENDIF
      END  !subroutine ges1
```

C***

A.19 itery.for

```
c--find which nuclear fraction is largest, and set the equations to
c--''pivot'' around that species.
C*******************************************************************
      SUBROUTINE itery(
     &y,ity)
       INTEGER ity
       REAL*8 y(7)
       IMPLICIT REAL*8 (a-h,l-z)
       ity=6
       IF (y(1).GT.y(6)) THEN
         ity=1
       ELSEIF (y(2).GT.y(1)) THEN
         ity=2
       ELSEIF (y(3).GT.y(2)) THEN
         ity=3
       ELSEIF (y(4).GT.y(3)) THEN
         ity=4
       ENDIF
       END  !subroutine itery
C*******************************************************************
```

A.20   getyh.for

```
c--get the sum of the nuclear fractions=the ''heavy'' fraction, yh.
C******************************************************************
      SUBROUTINE getyh(
     &y,yh)
      REAL*8 y(7),yh
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      yh=0.
      DO 100 i=1,7
100      yh=yh+y(i)
      END  !subroutine getyh
C******************************************************************
```

A.21 getye.for

c--get the electron fraction, ye, based on the charge weighted average
c--fraction

```
C****************************************************************
      SUBROUTINE getye(
     &y,ye)
      REAL*8 y(7),ye
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      ye=0.
      DO 100 i=1,5
100      ye=ye+(i-1)*y(i)
      ye=ye+y(7)
      END  !subroutine getye
C****************************************************************
```

A.22 getwyh.for

c--get the mass-weighted sum of the heavy nuclear fractions. Used for
c--conservation of mass equations.

```
C******************************************************************
      SUBROUTINE getwyh(
     &y,yAB,wyh)
      REAL*8 y(7),yAB(2),wyh
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      wyh=0.
      DO 100 i=1,5
100      wyh=wyh+y(i)
      wyh=wyh+2.*y(6)+2.*y(7)+yAB(1)+yAB(2)
      END !subroutine getwyh
C******************************************************************
```

A.23   balance.for

```
c--the inner loop.  Finishes when either (1) the mass weighted
c--nuclear fraction sum (wyh) is within epsilon of the total nuclear
c--fraction, y0, or (2) the sum of the heavy fractions (yhh, a
c--different name, so we don't screw up the outer loop) is very very
c--small.  This protects us against the case where all of an element
c--combines with the other to form the compound AB; otherwise, the routine
c--will keep trying to reduce the partial residual by making the
c--concentrations smaller and smaller, until finally we get an
c--underflow error.
C*******************************************************************
      SUBROUTINE balance(
     &ye,yh,beta,epsilon,chi,ymin,yAB,seein,y0,y,ity,rhs)
      INCLUDE format.for
      INTEGER ity,seein
      REAL*8 ye,yh,beta,epsilon,chi,ymin,yAB(2),y0,y(7),rhs(7)
      INTEGER iin
      IMPLICIT REAL*8 (a-h,l-z)
      loy=0.
      hiy=y0
      iin=0
      wyh=-1.
      yhh=1.
c-----the inner loop.
      DO WHILE ((DABS(wyh-y0).GT.epsilon).AND.(yhh.GT.ymin))
c-------get the new (''nu'') fractions (''y'')
        CALL nuy(ity,y0,y,ye,yh,beta,rhs)
c-------then get the new weighted heavy fraction
        CALL getwyh(y,yAB,wyh)
c-------under-relax/bisection the pivot y
        IF (wyh.GT.y0) THEN
          hiy=y(ity)
          y(ity)=(1.-chi)*y(ity)+chi*loy
        ELSEIF (wyh.LT.y0) THEN
          loy=y(ity)
          y(ity)=(1.-chi)*y(ity)+chi*hiy
        ENDIF
c-------get the new weighted heavy fraction with the updated pivot y
        CALL getwyh(y,yAB,wyh)
c-------and get the test-case heavy fraction
        CALL getyh(y,yhh)
        iin=iin+1
```

```
c-------how's the progress?
         IF (mod(iin,seein).EQ.0) THEN
           WRITE(*,192)'+',iin,wyh-y0
         ENDIF
      ENDDO
      END
C****************************************************************
```

A.24   nuy.for

```
c--given the pivot y (ity is the array index for the y array), the
c--total nuclear fraction y0, the temporary ye,yh,beta, and the rhs
c--array, we update the other nuclear fractions for an element.
C*******************************************************************
      SUBROUTINE nuy(
     &ity,y0,y,ye,yh,beta,rhs)
      INTEGER ity
      REAL*8 y0,y(7),ye,yh,beta,rhs(7)
      IMPLICIT REAL*8 (a-h,l-z)
c-----check if electron density is zero, and should still be zero.  Need
c-----to check, because otherwise, will try to divide by beta, which is
c-----zero if ye=0.
      IF (ye.EQ.0.) THEN
        y(2)=0.
        y(3)=0.
        y(4)=0.
        y(6)=0.
        y(7)=0.
c-------calculate the other non-charged species from the pivot species
      IF (ity.EQ.6) THEN
        y(1)=DSQRT(rhs(6)*y(6)*(y0-y(6)))
      ELSEIF (ity.EQ.1) THEN
        y(6)=2.*y(1)*y(1)/rhs(6)/(y0+y(1))
      ENDIF
c-----find the other nuclear fraction y's from the pivot y.  The
c-----equations are only slightly non-linear when we (temporarily) hold
c-----the electron fraction, ye, and beta (as well as yh) constant.
      ELSEIF (ye.GT.0.) THEN
        IF (ity.EQ.6) THEN
          y(1)=DSQRT((yh+ye)*rhs(6)*y(6))
          y(2)=rhs(1)/beta*y(1)
          y(3)=rhs(2)/beta*y(2)
          y(4)=rhs(3)/beta*y(3)
          y(5)=rhs(4)/beta*y(4)
          y(7)=rhs(7)/beta*y(6)
        ELSEIF (ity.EQ.1) THEN
          y(2)=y(1)*rhs(1)/beta
          y(3)=y(2)*rhs(2)/beta
          y(4)=y(3)*rhs(3)/beta
          y(5)=y(4)*rhs(4)/beta
          y(6)=y(1)*y(1)/rhs(6)/(ye+yh)
```

```fortran
      y(7)=y(6)*rhs(7)/beta
    ELSEIF (ity.EQ.2) THEN
      y(1)=y(2)*beta/rhs(1)
      y(3)=y(2)*rhs(2)/beta
      y(4)=y(3)*rhs(3)/beta
      y(5)=y(4)*rhs(4)/beta
      y(6)=y(1)*y(1)/rhs(6)/(ye+yh)
      y(7)=y(6)*rhs(7)/beta
    ELSEIF (ity.EQ.3) THEN
      y(2)=y(3)*beta/rhs(2)
      y(1)=y(2)*beta/rhs(1)
      y(4)=y(3)*rhs(3)/beta
      y(5)=y(4)*rhs(4)/beta
      y(6)=y(1)*y(1)/rhs(6)/(ye+yh)
      y(7)=y(6)*rhs(7)/beta
    ELSEIF (ity.EQ.4) THEN
      y(3)=y(4)*beta/rhs(3)
      y(2)=y(3)*beta/rhs(2)
      y(1)=y(2)*beta/rhs(1)
      y(5)=y(4)*rhs(4)/beta
      y(6)=y(1)*y(1)/rhs(6)/(ye+yh)
      y(7)=y(6)*rhs(7)/beta
    ENDIF
  ENDIF
  END  !subroutine nuy
C***********************************************************************
```

A.25   res1.for

```
c--get the residual for an element.  subroutine can be used for either
c--element; add residuals from both elements, and hetero-diatomics to
c--get total residual.
C**********************************************************************
      SUBROUTINE res1(
     &y0,y,rhs,yAB,ye,yh,beta,residual)
      REAL*8 y0,y(7),rhs(7),yAB(2),ye,yh,beta,residual
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      REAL*8 res(7)
      CALL getwyh(y,yAB,wyh)
      res(1)= wyh-y0
      res(2)=-rhs(1)*y(1)+beta*y(2)
      res(3)=-rhs(2)*y(2)+beta*y(3)
      res(4)=-rhs(3)*y(3)+beta*y(4)
      res(5)=-rhs(4)*y(4)+beta*y(5)
      res(6)= y(1)*y(1)/(yh+ye)-rhs(6)*y(6)
      res(7)=-rhs(7)*y(6)+beta*y(7)
      residual=0.
      DO 100 i=1,7
        residual=residual+DABS(res(i))
100   CONTINUE
      END  !subroutine res1
C**********************************************************************
```

A.26   res2.for

```fortran
c--get residual for heteronuclear diatomic species.
C*****************************************************************
      SUBROUTINE res2(
     &yA,yB,yAB,rhsAB,ye,yh,beta,residual)
      REAL*8 yA(7),yB(7),yAB(2),rhsAB(2),ye,yh,beta,residual
      INTEGER i
      IMPLICIT REAL*8 (a-h,l-z)
      REAL*8 res(2)
      res(1)= yA(1)*yB(1)/(yh+ye)-rhsAB(1)*yAB(1)
      res(2)=-rhsAB(2)*yAB(1)+beta*yAB(2)
      residual=0.
      DO 100 i=1,2
         residual=residual+DABS(res(i))
100   CONTINUE
      END  !subroutine res2
C*****************************************************************
```

A.27   gas.for

```
C**********************************************************************
      SUBROUTINE gas(
     &press1,temp,
     &DA2,IA2,ionA,yA,enthA,CpA,entrA,
     &DB2,IB2,ionB,yB,enthB,CpB,entrB,
     &name,RAB,IAB,yAB,enthAB,CpAB,entrAB,enthe,Cpe,entre,
     &enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &dirA,dirB,dirAB)
      CHARACTER name*4
      REAL*8
     &press1,temp,
     &DA2,IA2,ionA(5),yA(7),enthA(7),CpA(7),entrA(7),
     &DB2,IB2,ionB(5),yB(7),enthB(7),CpB(7),entrB(7),
     &RAB,IAB,yAB(2),enthAB(2),CpAB(2),entrAB(2),
     &enthe,Cpe,entre,
     &enthal,heatcp,entrop,mfA(7),mfB(7),mfAB(2),
     &denA(7),denB(7),denAB(2),dirA(7),dirB(7),dirAB(2)
      INTEGER i
      IMPLICIT REAL*8 (a-h,1-z)
      DATA boltzk/1.38066E-23/
      DATA JmoleV,Jmolcm/96484.52,11.96263/
c-----JmoleV (96484.52 J/mol/eV) converts eV to J/mol
c-----re-evaluate the heavy and electron fractions, just t-  '. safe.
      CALL getye(yA,yeA)
      CALL getye(yB,yeB)
      CALL getyh(yA,yhA)
      CALL getyh(yB,yhB)
      ye=yeA+yeB+yAB(2)
      yh=yhA+yhB+yAB(1)+yAB(2)
      pkt=press1/(boltzk*temp*(ye+yh))
      enthal=0.
      heatcp=0.
      entrop=0.
c-----calculate the heats of formation for the species:
c-----dissociation (''d''), ionization (''i''), and reactions (''r'')
c-----also calculate the gas thermodynamic properties, based on the
c-----species properties and the mole fractions.
      dirA(1)=0.5*DA2*JmoleV    !J/mol
      dirB(1)=0.5*DB2*JmoleV
      DO 100 i=1,5
         enthal=enthal+yA(i)*enthA(i)+yB(i)*enthB(i)
```

```
            heatcp=heatcp+yA(i)*CpA(i)+yB(i)*CpB(i)
            entrop=entrop+yA(i)*entrA(i)+yB(i)*entrB(i)
            dirA(i+1)=dirA(i)+ionA(i)*Jmolcm    !J/mol
            dirB(i+1)=dirB(i)+ionB(i)*Jmolcm
            mfA(i)=yA(i)/(yh+ye)   !mole fraction
            mfB(i)=yB(i)/(yh+ye)
            denA(i)=yA(i)*pkt
            denB(i)=yB(i)*pkt
100     CONTINUE
        DO 200 i=1,2
            enthal=enthal+
     &  yAB(i)*enthAB(i)+yA(i+5)*enthA(i+5)+yB(i+5)*enthB(i+5)
            heatcp=heatcp+
     &  yAB(i)*CpAB(i)+yA(i+5)*CpA(i+5)+yB(i+5)*CpB(i+5)
            entrop=entrop+
     &  yAB(i)*entrAB(i)+yA(i+5)*entrA(i+5)+yB(i+5)*entrB(i+5)
            mfA(i+5)=yA(i+5)/(yh+ye)
            mfB(i+5)=yB(i+5)/(yh+ye)
            denA(i+5)=yA(i+5)*pkt
            denB(i+5)=yB(i+5)*pkt
            mfAB(i)=yAB(i)/(yh+ye)
            denAB(i)=yAB(i)*pkt
200     CONTINUE
        dirA(6)=0.
        dirB(6)=0.
        dirA(7)=IA2*JmoleV
        dirB(7)=IB2*JmoleV
        dirAB(1)=(0.5*(DA2+DB2)-RAB)*JmoleV
        dirAB(2)=(0.5*(DA2+DB2)-RAB+IAB)*JmoleV
        enthal=enthal+ye*enthe
        heatcp=heatcp+ye*Cpe
        entrop=entrop+ye*entre
        enthal=enthal/(ye+yh)
        heatcp=heatcp/(ye+yh)
        entrop=entrop/(ye+yh)
c        WRITE(*,'(A,1PE11.3)')' Enthalpies (J/mol)'//
c     &name//' gas',enthal
c        WRITE(*,'(A,1PE11.3)')' Heat capacities (c_p) (J/K/mol)'//
c     &name//' gas',heatcp
        END !subroutine output
C***********************************************************************
```

A.28   rocket.for

```
c--get performance from an ideal rocket, assuming (1) our
c--temperature and pressure are that of the rocket chamber, (2) frozen
c--flow (no change in equilibrium species composition downstream), (3)
c--isentropic (reversible) expansion of gases, (4) hard sphere ideal gas
c--(so forget all the complicated thermodynamic functions we calculated;
c--just use c_p=2.5R).  Frozen flow assumption relieves us from having
c--to calculate coronal equilbrium at exit conditions.
C************************************************************************
      SUBROUTINE rocket(
     &yOA,yOB,MWA,MWB,ye,yh,temp,press1,press2,mdot,thrust,Isp,Pjet)
      REAL*8 yOA,yOB,MWA,MWB,ye,yh,
     &temp,press1,press2,mdot,thrust,Isp,Pjet
      REAL*8 MWbar,prat,vex
      DATA gasR,grav/8.31439,9.8062/
      MWbar=(yOA*MWA+yOB*MWB)/(yh+ye)
      prat=press2/press1
      vex=DSQRT(5.e3*gasR/MWbar*temp*(1-prat*prat*DSQRT(prat)))
      thrust=mdot*vex
      Isp=vex/grav
      Pjet=0.5*mdot*vex*vex
      END !subroutine rocket
C************************************************************************
```

A.29　output.for

```
c--The mother of all subroutines.  Basically has two branches: (1) if
c--the has selected howmany=1, this subroutine will print anything of
c--interest to a file name name.out, where ''name'' is the name of the
c--compound formed by the two species the user chose.  The user gets his
c--original input data, the electronic, rotational, and vibrational
c--partition functions of all the species, the right hand sides of the
c--saha equation, the nuclear fractions (y's) of the species, the mole
c--fractions of the species, the densities, the heats of formation for
c--the species, the enthalpies, heat capacities, and entropies, both of
c--the species and for the entire gas, and the rocket performance.  If
c--you want it and its not there, this program doesn't do it. (2) If the
c--user selects multiple temperatures (howmany.!= 1), a more
c--convenient, tabulated form of data is outputted to several files,
c--one file per data type per element.  However, due to the MS-DOS
c--limitations on the number of possible files open at one time, the
c--user must choose between density and mole fraction.  I chose these
c--since it's not likely the user will want both.  Depending on the
c--user's choice of howmany, subSUBROUTINEs open the appropriate files
c--and make the appropriate headers.
C******************************************************************
      SUBROUTINE output(
     &yOA,yOB,temp1,temp2,howmany,press1,press2,mdot,temp,itemp,
     &chA,ZA,rhsA,yA,enthA,CpA,entrA,
     &chB,ZB,rhsB,yB,enthB,CpB,entrB,
     &name,ZAB,rhsAB,yAB,enthAB,CpAB,entrAB,enthe.Cpe,entre,
     &enthal,heatcp,entrop,mfA,mfB,mfAB,denA,denB,denAB,
     &dirA,dirB,dirAB,Pabs,Pjet,thrust,Isp,densmol)
      CHARACTER chA*2,chB*2,name*4
      INTEGER itemp,densmol,howmany
      REAL*8
     &yOA,yOB,temp1,temp2,press1,press2,mdot,temp,
     &ZA(7,3),rhsA(7),yA(7),enthA(7),CpA(7),entrA(7),
     &ZB(7,3),rhsB(7),yB(7),enthB(7),CpB(7),entrB(7),
     &ZAB(2,3),rhsAB(2),yAB(2),enthAB(2),CpAB(2),entrAB(2),
     &enthe,Cpe,entre,
     &enthal,heatcp,entrop,mfA(7),mfB(7),mfAB(2),
     &denA(7),denB(7),denAB(2),dirA(7),dirB(7),dirAB(2),
     &Pabs,Pjet,thrust,Isp
      CHARACTER filename*12
      IMPLICIT REAL*8 (a-h,l-z)
      DATA boltzk/1.38066E-23/
```

```fortran
      INCLUDE format.for
c-----re-evaluate the heavy and electron fractions, just to be safe.
      CALL getye(yA,yeA)
      CALL getye(yB,yeB)
      CALL getyh(yA,yhA)
      CALL getyh(yB,yhB)
      ye=yeA+yeB+yAB(2)
      yh=yhA+yhB+yAB(1)+yAB(2)
      beta=ye/(yh+ye)
      pkt=press1/(boltzk*temp*(ye+yh))
c-----send a teaser to the screen
      WRITE(*,192)' Pabs, Pjet (W) ',Pabs, Pjet
      WRITE(*,192)' Thrust(N),Isp(s)',thrust,Isp
      IF (howmany.EQ.1) THEN
c-----output for single input
         IF (itemp.EQ.1) THEN
           OPEN (UNIT=90, FILE=filename(name,'.out'))
         ENDIF
c-------write input info
         WRITE(90,'(A)')'Using '//chA//'',','//chB//'','//name
         WRITE(90,*)'Stoichiometric Proportions of '//chA//'',''//chB
         WRITE(90,'('' ['',1PE11.3,'','',1PE11.3,''] '')')y0A,y0B
         WRITE(90,*)'chamber, exit pressures (atm)'
         WRITE(90,'('' ['',1PE11.3,'','',1PE11.3,''] '')')press1,press2
         WRITE(90,*)'Temperature range: temp1(K),temp2(K),howmany'
         WRITE(90,'('' ['',1PE11.3,'','',1PE11.3,'','',I5,''] '')')
     &   temp1,temp2,howmany
         WRITE(90,*)'mass flow rate (kg/s)'
         WRITE(90,'('' ['',1PE11.3,''] '')')mdot
c-------pretty display of partition function
         CALL Z1show(chA,ZA)
         CALL Z1show(chB,ZB)
         CALL Z2show(name,ZAB)
c-------show the information of a species, described by the first
c-------parameter string
         CALL specshow('Right-hand-sides of Saha Equation',
     &   chA,chB,name,rhsA,rhsB,rhsAB)
         CALL specshow('Nuclear fractions,y',
     &   chA,chB,name,yA,yB,yAB)
         WRITE(90,194)'yh,ye,yh+ye,beta',yh,ye,yh+ye,beta
         CALL specshow('Mole fractions,y/(yh+ye)',
     &   chA,chB,name,mfA,mfB,mfAB)
```

```
            WRITE(90,191)'e-',ye/(yh+ye)
            CALL specshow('Species densities (m*-3)',
     &   chA,chB,name,denA,denB,denAB)
            WRITE(90,191)'e-',pkt*ye
            CALL specshow('Heat of formation (J/mol)',
     &   chA,chB,name,dirA,dirB,dirAB)
            CALL specshow('Enthalpies (J/mol)',
     &   chA,chB,name,enthA,enthB,enthAB)
            WRITE(90,191)'e-',enthe
            WRITE(90,191)name//' gas',enthal
            CALL specshow('Heat capacities (C_p) (J/K/mol)',
     &   chA,chB,name,CpA,CpB,CpAB)
            WRITE(90,191)'e-',Cpe
            WRITE(90,191)name//' gas',heatcp
            CALL specshow('Entropy diff (from STP) (J/K/mol)',
     &   chA,chB,name,CpA,CpB,CpAB)
            WRITE(90,191)'e-',entre
            WRITE(90,191)name//' gas',entrop
            WRITE(90,*)
            WRITE(90,192)'Thrust(N),Isp(s)',thrust,Isp
            WRITE(90,192)'Pabs, Pjet (W)',Pabs,Pjet
        ELSE
c-----output for multiple inputs
            IF (itemp.EQ.1) THEN
c---------open all the appropriate files for output
            CALL opunhdr(chA,chB,name,densmol)
            ENDIF
c-------write all the data to all the files
            WRITE(61,139)temp,enthA,enthe
            WRITE(62,139)temp,enthB,enthe
            WRITE(63,134)temp,enthAB,enthe
            WRITE(71,139)temp,CpA,Cpe
            WRITE(72,139)temp,CpB,Cpe
            WRITE(73,134)temp,CpAB,Cpe
            WRITE(81,139)temp,entrA,entre
            WRITE(82,139)temp,entrB,entre
            WRITE(83,134)temp,entrAB,entre
c-------the MS-DOS-imposed choice
            IF (densmol.EQ.1) THEN
              WRITE(91,139)temp,denA,pkt*ye
              WRITE(92,139)temp,denB,pkt*ye
              WRITE(93,134)temp,denAB,pkt*ye
```

```fortran
        ELSE
          WRITE(91,139)temp,mfA,ye/(yh+ye)
          WRITE(92,139)temp,mfB,ye/(yh+ye)
          WRITE(93,134)temp,mfAB,ye/(yh+ye)
        ENDIF
        WRITE(94,135)temp,Pabs,Pjet,thrust,Isp
      ENDIF
      END !subroutine output
C*********************************************************************
```

A.30   opunhdr.for

```
c--opens (''opun'') all the output files for the howmany!=1 case, and
c--writes the headers (''hdr''), such as a line describing what the file
c--contains, with units, and species column headings.
c--file unit key
c--60's=enthalpy
c--70's=heat capcity
c--80's=entropy
c--90's=density/molefraction
c--+1=element A
c--+2=element B
c--+3=hetero-dia molecule AB
c--output file naming convention is as follows:
c--(name)_(description).out
c--where (name) is the element or hetero-dia molecule name
c--and (description) is {eth,cp,etr,den,mol} for enthalpy, etc.
C********************************************************************
      SUBROUTINE opunhdr(
     &chA,chB,name,densmol)
       CHARACTER chA*2,chB*2,name*4,
     &filename*12,prefix*4(3),exten*8(4),header*40(4)
       INTEGER densmol,i,j,ijunit
       prefix(1)=chA
       prefix(2)=chB
       prefix(3)=name
       exten(1)='_eth.out'
       exten(2)='_Cp.out'
       exten(3)='_etr.out'
       header(1)='Enthalpy (J/mol)'
       header(2)='Heat capacity, C_p (J/K/mol)'
       header(3)='Entropy difference from STP (J/K/mol)'
       IF (densmol.EQ.1) THEN
         exten(4)='_den.out'
         header(4)='Partial density (m^-3)'
       ELSE
         exten(4)='_mol.out'
         header(4)='Mole fraction'
       ENDIF
       DO 10 i=1,4
         DO 10 j=1,3
           ijunit=60+10*(i-1)+j
           OPEN (UNIT=ijunit,FILE=filename(prefix(j),exten(i)))
```

```fortran
10      CONTINUE
        OPEN (UNIT=94, FILE=filename(name,'_rkt.out'))
        DO 20 i=1,4
          DO 20 j=1,3
            ijunit=60+10*(i-1)+j
20          WRITE(ijunit,*)prefix(j),header(i)
        WRITE(94,*)name,' rocket performance parameters'
991     FORMAT(2x,'temp(K)',4x,A,9x,A,'+',8x,A,'+2',7x,A,'+3',7x,A,
       &'+4',7x,A,'2',8x,A,'2+',7x,'e-')
993     FORMAT(2x,'temp(K)',4x,A,7x,A,'+',6x,'e-')
        DO 30 i=1,4
          ijunit=60+10*(i-1)
          WRITE(ijunit+1,991)chA,chA,chA,chA,chA,chA,chA
          WRITE(ijunit+2,991)chB,chB,chB,chB,chB,chB,chB
          WRITE(ijunit+3,993)name,name
30      CONTINUE
994     FORMAT(2x,'temp(K)',4x,'Pabs(W)',4x,'Pjet(W)',4x,'thrust(N)',
       &2x,'Isp(s)')
        WRITE(94,994)
        END !opunhdr
C*******************************************************************
```

A.31 filename.for

c--takes two strings (of arbitrary length) and concatenates them, taking
c--the spaces out of the first string. This is particularly useful when
c--making a filename for the OPEN command that MS-DOS will recognize.
c--If there are spaces in between the prefix and the extension, MS-DOS
c--makes a file with no extension...

```
C*****************************************************************
      CHARACTER*12 FUNCTION filename(name,exten)
      CHARACTER name*(*),exten*(*),temp*12
      INTEGER ilast
      temp=name
      ilast=INDEX(name,' ')
      IF (ilast.eq.0) THEN
        temp(LEN(name)+1:)=exten
      ELSE
        temp(ilast:)=exten
      ENDIF
      filename=temp
      END
C*****************************************************************
```

A.32 specshow.for

```
c--displays the contents of a species variable to the howmany=1 file,
c--but in an user-interpretable fashion.
C*****************************************************************************
      SUBROUTINE specshow(
     &str,chA,chB,name,yA,yB,yAB)
      CHARACTER str*(*),chA*2,chB*2,name*4
      REAL*8 yA(7),yB(7),yAB(2)
100   FORMAT (4x,5('atomic,',4x),2('diatomic,',2x))
101   FORMAT (4x,'neutral',4x,4('+',I1,9x),'neutral',4x,'+1')
102   FORMAT (A,1P7(E11.3))
103   FORMAT (A,53x,1P2(E11.3))
      WRITE(90,*)
      WRITE(90,*)str
      WRITE(90,100)
      WRITE(90,101)1,2,3,4
      WRITE(90,102)chA,yA
      WRITE(90,102)chB,yB
      WRITE(90,103)name,yAB
      END
C*****************************************************************************
```

A.33   z1show.for

```
c--shows the partition function for an element in an intelligible way.
C******************************************************************
      SUBROUTINE Z1show(
     &ch,Z1)
      CHARACTER ch*2,partstr*3(3)
      REAL*8 Z1(7,3)
      INTEGER i
      DATA partstr/'el ','vib','rot'/
100   FORMAT (4x,5('atomic,',4x),2('diatomic,',2x))
101   FORMAT (4x,'neutral',4x,4('+',I1,9x),'neutral',4x,'+1')
      WRITE(90,*)
      WRITE(90,*)'Partition Function: ',ch
      WRITE(90,100)
      WRITE(90,101)1,2,3,4
      DO 10 i=1,3
        WRITE(90,'(A,1PE10.3,1P6(E11.3))')
     &  partstr(i),Z1(1,i),Z1(2,i),Z1(3,i),
     &  Z1(4,i),Z1(5,i),Z1(6,i),Z1(7,i)
10    CONTINUE
      END
C******************************************************************
```

A.34 z2show.for

c--shows the hetero-diatomic species partition function; for use with
c--the howmany=1 output file.

```
C********************************************************************
      SUBROUTINE Z2show(
     &name,Z2)
      CHARACTER name*4,partstr*3(3)
      REAL*8 Z2(7,3)
      INTEGER i
      DATA partstr/'el ','vib','rot'/
100   FORMAT (4x,5('atomic,',4x),2('diatomic,',2x))
101   FORMAT (4x,'neutral',4x,4('+',I1,9x),'neutral',4x,'+1')
      WRITE(90,*)
      WRITE(90,*)'Partition Function: ',name
      WRITE(90,100)
      WRITE(90,101)1,2,3,4
      DO 10 i=1,3
        WRITE(90,'(A,54x,1P2(E11.3))')partstr(i),Z2(1,i),Z2(2,i)
10    CONTINUE
      END
C********************************************************************
```

A.35   shutdown.for

```
c--shuts down the program, closing all the open files, telling the user
c--which files to look for, and waiting for an <Enter> before returning
c--control to DOS.  This helps when running the program from a DOS
c--shell, like DCOM, which may automatically go back to the interface
c--screen and prevent the user from seeing which files he needs to see.
C***************************************************************************
      SUBROUTINE shutdown(
     &howmany,chA,chB,name,densmol)
      CHARACTER chA*2,chB*2,name*4,
     &filename*12,prefix*4(3),exten*8(4),pause*1
      INTEGER densmol,i,j,howmany
      IF (howmany.EQ.1) THEN
        WRITE(*,*)'Look for output in file (',filename(name,'.out'),')'
      ELSE
        prefix(1)=chA
        prefix(2)=chB
        prefix(3)=name
        exten(1)='_eth.out'
        exten(2)='_Cp.out'
        exten(3)='_etr.out'
        IF (densmol.EQ.1) THEN
          exten(4)='_den.out'
        ELSE
          exten(4)='_mol.out'
        ENDIF
        WRITE(*,*)'Look for output in files:'
        DO 10 i=1,4
          DO 10 j=1,3
            WRITE(*,*)'(',filename(prefix(j),exten(i)),')'
10      CONTINUE
        WRITE(*,*)'(',filename(name,'_rkt.out'),')'
      ENDIF
      CLOSE(UNIT=61)
      CLOSE(UNIT=62)
      CLOSE(UNIT=63)
      CLOSE(UNIT=71)
      CLOSE(UNIT=72)
      CLOSE(UNIT=73)
      CLOSE(UNIT=81)
      CLOSE(UNIT=82)
      CLOSE(UNIT=83)
```

```
      CLOSE(UNIT=90)
      CLOSE(UNIT=91)
      CLOSE(UNIT=92)
      CLOSE(UNIT=93)
      WRITE(*,*)'Press <Enter> to exit program...'
      READ(*,'(A)')pause
      END !subroutine shutdown
C***********************************************************************
```

```
102    FORMAT (2I4)
111    FORMAT (A,I4)
112    FORMAT (A,2I4)
113    FORMAT (A,3I4)
131    FORMAT (1P1(E11.3))
132    FORMAT (1P2(E11.3))
133    FORMAT (1P3(E11.3))
134    FORMAT (1P4(E11.3))
135    FORMAT (1P5(E11.3))
136    FORMAT (1P6(E11.3))
137    FORMAT (1P7(E11.3))
138    FORMAT (1P8(E11.3))
139    FORMAT (1P9(E11.3))
1310   FORMAT (1P10(E11.3))
1311   FORMAT (1P11(E11.3))
1312   FORMAT (1P12(E11.3))
1313   FORMAT (1P13(E11.3))
1314   FORMAT (1P14(E11.3))
1315   FORMAT (1P15(E11.3))
1316   FORMAT (1P16(E11.3))
1317   FORMAT (1P17(E11.3))
1318   FORMAT (1P17(E11.3))
191    FORMAT (A,1PE11.3)
192    FORMAT (A,1P2(E11.3))
193    FORMAT (A,1P3(E11.3))
194    FORMAT (A,1P4(E11.3))
195    FORMAT (A,1P5(E11.3))
196    FORMAT (A,1P6(E11.3))
197    FORMAT (A,1P7(E11.3))
198    FORMAT (A,1P8(E11.3))
199    FORMAT (A,1P9(E11.3))
1910   FORMAT (A,1P10(E11.3))
1911   FORMAT (A,1P11(E11.3))
1912   FORMAT (A,1P12(E11.3))
1913   FORMAT (A,1P13(E11.3))
1914   FORMAT (A,1P14(E11.3))
1915   FORMAT (A,1P15(E11.3))
1916   FORMAT (A,1P16(E11.3))
```

B   Input Files

B.1   abx.in

ist	MW(amu)	uD(eV)	uI(eV)	Tr,0(K)	Tv,0(K)	Tr,1(K)	Tv,1(K)
C 5	12.011	6.21	12.1	2.618	2668.	2.386	1942.
H 2	1.0079	4.47	15.4	87.55	6332.	43.45	3339.
N 4	14.0067	9.75	15.5	2.875	3393.	2.779	3175.
O 5	15.9994	5.11	12.0	2.079	2273.	2.433	2740.
Ar 5	1.800E+01	-5.000E+01	5.000E+01	1.000E+00	1.000E+00	1.000E+00	1.000E+00
D 2	2.0	4.47	15.4	87.55	6332.	43.45	3339.

B.2   abxy.in

	uD(eV)	uI(eV)	Tr,0(K)	Tv,0(K)	Tr,1(K)	Tv,1(K)
CC	6.21	12.1	2.618	2668.	2.386	1942
CH	3.46	10.6	20.80	4112.	20.39	3941
CN	7.70	14.1	2.733	2976.	2.728	2925
CO	11.0	14.0	2.778	3121.	2.844	3185
HH	4.47	15.4	87.55	6332.	43.45	3339
HN	3.47	13.6	24.02	4722.	22.08	4204
HO	4.39	12.9	27.20	5377.	24.16	4479
NN	9.75	15.5	2.875	3393.	2.779	3175
NO	6.49	9.26	2.405	2739.	2.873	3419
OO	5.11	12.0	2.079	2273.	2.433	2740
ArD	-5.000E+01	5.000E+01	1.000E+00	1.000E+00	1.000E+00	1.000E+00

B.3   ar-0.in

```
127109.9
1 0
5 93143.8
3 93750.639
1 94553.707
3 95399.870
3 104102.144
7      105462.804
5      105617.315
3      106087.305
5      106237.597
1      107054.319
3      107131.755
5      107289.747
3      107496.463
1      108722.668
1      111667.87
3      111818.09
9      112750.22
7      113020.39
5      112138.98
3      114147.75
5      113426.05
7      113716.61
5      114641.04
7      114821.99
5      114805.18
3      115366.90
5      113468.55
3      113643.26
1      114861.67
3      114975.07
3      116660.054
7      116942.815
5      116999.389
3      117151.387
5      117183.654
1      117563.020
3      118407.494
5      118469.117
3      118459.662
```

1	118870.981
1	118512.17
3	118651.447
9	119023.699
7	119212.93
5	118906.665
3	119847.81
5	119444.88
7	119566.11
5	120619.076
7	120753.52
5	120600.944
3	121011.979
5	119683.113
3	119760.22
1	121096.67
3	121161.356
3	120188.34
5	120188.66
11	120207.32
9	120207.77
7	120229.81
5	120230.07
16	120250.15
16	121653.40
7	121654.32
5	121654.58
3	121068.804
7	121165.431
5	121191.92
3	121257.227
5	121270.682
1	121470.304
3	122609.76
5	122635.128
3	122601.290
1	122790.612
1	121794.158
3	121932.908
9	122036.134
7	122160.22
5	122086.974

3	122514.29
5	122282.134
7	122329.72
5	123505.536
7	123557.459
5	123372.987
3	123815.53
5	122440.109
3	122479.459
1	123873.07
3	123882.30
3	122686.20
5	122686.40
20	122695.70
7	122707.94
5	122708.18
16	122717.90
16	124135.74
7	124137.29
5	124137.45
3	123172.09
7	123205.83
5	123220.73
3	123254.99
5	123261.593
1	123385.13
3	124643.54
5	124658.52
3	124651.05
1	124749.89
1	123508.96
3	123468.034
9	123653.238
7	123773.920
8	123808.60
5	123826.85
7	123832.50
5	125113.48
7	125150.00
5	125066.501
3	125286.28
5	123903.295

3	123935.97
1	125334.75
3	125353.31
3	124041.20
5	124041.38
20	124046.64
7	124051.44
5	124051.65
16	124058.36
16	125482.70
7	125483.16
5	125483.34
3	124311.72
7	124349.04
5	124356.73
3	124376.38
5	124381.01
1	124439.41
3	125783.8
5	125791.94
3	125777.3
1	125831.45
1	124526.75
3	124554.939
9	124609.917
7	124649.549
5	124603.957
3	124788.39
5	124692.02
7	124715.16
5	126064.50
7	126089.56
8	126053.21
5	124771.67
3	124782.77
1	126202.82
3	126211.57
3	124857.27
5	124857.42
20	124860.64
7	124865.04
5	124865.19

```
16      124868.77
16      126294.90
12      126295.02
3       125039.60
7       125054.1
5       125059.8
3       125072.6
5       125074.9
1       125122.54
4       126524.2
1       125163.00
3       125135.898
9       125219.88
7       125269.52
8       125282.97
5       125291.45
7       125293.65
5       125329.99
3       125331.93
8       125386.41
20      125388.65
7       125391.04
5       125391.17
16      125393.79
3       125505.5
12      125519.9
3       125531.5
5       125533.8
1       125561.9
1       125595.11
3       125613.12
-1      -1
```

B.4 ar-1.in

```
222820.
4 0.
2 1432.0
2 108722.5
8 132328.22
6 132482.12
4 132631.64
2 132738.60
6 134242.62
4 135086.88
2 135602.62
4 138244.51
2 139259.22
10 142187.42
8 142718.01
6 143108.63
4 143372.48
2 144710.90
4 145669.84
2 147229.17
4 147504.12
6 147876.98
4 148620.98
6 148843.29
8 149180.18
6 150148.54
4 150475.82
6 151088.18
6 155044.07
4 155352.04
2 155709.02
8 157234.93
6 157674.30
4 158168.71
2 158429.05
6 158731.2
4 159394.32
2 159707.46
4 160240.35
4 161049.65
2 161090.31
```

```
 2 167308.66
 6 170401.88
 8 170531.29
 4 172214.74
 2 172817.12
 6 172336.47
 4 172830.63
 4 173348.78
 6 173394.33
 4 174410.74
 2 174821.94
 4 179593.09
 2 179932.83
 6 181595.04
 4 182223.06
 2 182952.14
 4 183091.83
 2 183915.58
 8 183676.42
 6 183798.22
 4 183986.83
 2 184193.12
 2 184094.10
10 185093.92
 8 185625.47
 6 186074.06
 4 186341.39
 2 186172.32
 4 186471.32
 6 186891.92
 6 186728.28
 4 186750.78
 8 186817.12
 6 187589.62
 2 189935.62
 4 190593.62
 4 190106.84
 2 190196.80
10 190508.00
 2 191708.46
 4 191975.16
 2 192334.09
```

```
6 192557.77
4 192712.93
10 194800.97
8 194822.95
6 194862.31
4 194997.65
8 194883.96
6 195032.13
4 195298.62
2 195282.50
6 195865.61
4 195867.73
4 196077.40
2 196091.04
4 196622.78
6 196633.93
8 198595.91
10 198604.78
6 198813.17
4 199138.92
2 200111.16
2 199447.56
4 199982.96
4 199525.96
6 199680.58
4 200032.65
2 200624.00
8 200139.84
6 200235.70
4 204418.50
2 204515.81
10 204586.40
2 205243.96
6 208592.90
4 212932.88
6 212934.30
-1 -1
```

B.5   ar-2.in

```
329965.80
5 0.
3 1112.40
1 1570.20
5 14010.
1 33267.
5 113800.70
3 114797.60
1 115328.40
3 144023.
4 144882.93
5 144885.97
7 144892.95
9 144907.00
7 156917.62
5 156924.68
3 157031.40
5 174375.00
3 180679.00
9 186402.15
7 186657.20
5 186903.05
3 187171.12
5 187823.05
7 188714.05
9 188517.32
3 196589.20
5 196613.91
7 196679.80
3 204563.53
5 204649.24
7 204797.37
3 204727.47
5 207233.09
3 207532.15
1 207673.16
5 209151.82
3 209127.04
1 209166.35
7 210212.26
5 211004.85
```

```
3 211563.83
5 213950.87
3 214346.70
1 214568.49
3 225155.18
5 225147.93
7 225402.59
5 226355.96
7 226503.22
9 226646.06
5 231341.80
3 231627.30
1 231754.80
3 239193.48
3 240150.66
5 240257.59
7 240291.66
1 242923.96
3 243145.76
5 243424.97
4 246029.76
5 246033.79
7 246036.64
9 246046.57
5 250712.27
3 252272.92
5 252253.69
7 252289.02
3 252575.88
5 266722.80
7 266877.50
9 267071.22
7 267782.10
9 267833.20
11 267895.82
3 268978.80
5 269012.80
7 269000.80
5 271507.88
3 271672.08
1 271696.22
3 272068.45
```

```
3 272127.82
5 272188.16
7 272250.90
5 281461.97
16 281473.82
1 281947.88
3 282000.26
5 282099.14
7 283919.78
5 284096.26
3 284118.51
1 285831.20
3 285882.00
5 286009.21
-1 -1
```

B.6   ar-3.in

```
482400.
4 0.
4 21090.
6 21219.
2 34854.
4 35035.
6 117564.
4 118515.
2 119044.
4 145921.
6 146000.
4 166356.
2 167444.
2 177833.
2 250219.45
4 250906.60
6 251972.00
2 256093.29
4 257348.89
6 268151.38
4 268171.38
2 285960.17
4 286228.80
6 286751.68
8 287555.83
2 289125.88
4 289237.82
6 289834.68
4 290256.45
6 291667.73
4 291748.70
2 295674.54
4 295806.77
2 299563.20
6 304074.29
8 304399.90
6 306236.28
4 306308.25
-1 -1
```

B.7 ar-4.in

```
605100.
1 0.
3 765.
5 2032.
5 16301.
3 121632.
5 121678.
7 121840.
5 141764.
4 141773.
3 191537.
3 195356.
5 217578.
3 218286.
1 218642.
3 224216.
5 224505.
7 224717.
1 295742.
3 296249.
5 297893.
3 301300.
-1 -1
```

B.8 c-0.in

```
90878.3
1 0.0
3 16.4
5 43.5
5 10193.70
1 21648.4
5 33735.2
1 60333.80
3 60353.00
5 60393.52
3 61982.20
7 64088.56
5 64093.19
3 64092.01
3 68858.
3 69689.79
5 69710.99
7 69744.40
3 70744.26
1 71352.81
3 71365.23
5 71385.70
5 72611.06
1 73976.23
9 75256.3
5 77680.5
1 78105.23
3 78117.06
5 78148.36
5 78199.34
7 78215.82
9 78250.22
3 78300.8
5 78307.
7 78316.
3 78338.
7 78531.
3 78727.91
5 79311.10
3 79319.06
1 79323.32
```

3 80173.29
5 80192.49
7 80222.74
3 80563.57
3 81105.70
1 81311.52
3 81326.33
5 81344.48
5 81770.36
1 82252.31
5 83500.
7 83761.
3 83830.
5 83837.
7 83847.
3 83882.5
7 83949.
3 84032.
5 84102.6
3 84112.
3 84852.13
5 84952.
7 84986.2
5 85400.38
1 85625.84
5 86187.
5 86319.
7 86326.7
5 86371.3
7 86396.
3 86413.96
7 86450.
3 86491.
5 86504.
3 86517.
5 87632.
5 87706.
7 87713.
5 87752.
7 87773.
3 87795.3
7 87807.

```
5 87830.
3 87839.
3 87831.3
5 88541.8
7 88547.
7 88607.
7 88624.
3 88632.44
5 88639.
7 89081.
5 89082.
7 89146.
7 89155.
5 89158.
5 89450.
7 89514.
7 89517.
7 89779.
7 89968.4
-1 -1
```

B.9   c-1.in

```
196659.0
2 0.0
4 64.0
1 46000.2
4 43021.8
6 43050.7
6 74930.9
4 74933.2
2 96494.1
2 110625.1
4 110666.3
2 116537.88
2 131724.68
4 131735.81
4 142024.4
4 145549.99
6 145551.44
6 150462.8
4 150467.9
2 157234.43
2 162518.70
4 162524.62
2 166964.70
4 166988.46
6 167033.43
4 168123.92
6 168124.33
2 168731.6
4 168750.2
14 168979.05
2 173348.18
2 175287.9
4 175295.2
2 178194.1
4 178220.8
6 178494.8
14 178956.46
2 181258.
2 181694.50
4 181709.20
6 181734.21
```

```
8 181770.48
2 182025.0
4 182044.5
6 184064.9
14 184376.20
4 184688.69
2 186425.02
4 186441.32
6 186463.75
4 188579.3
6 188612.7
2 194571.9
4 195750.8
6 195765.1
8 195784.7
10 195812.3
2 196556.2
4 196561.8
6 196570.5
8 196580.8
-1 -1
```

B.10   c-2.in

```
386159.7
1 0.0
1 52315.0
3 52338.0
5 52394.8
3 102351.4
1 137374.0
3 137403.4
5 137450.5
5 145875.1
1 182520.2
3 238160.7
1 247169.5
3 258931.4
1 259653.8
3 259659.3
5 259672.1
3 269957.6
5 269959.7
7 269962.9
5 276482.7
1 308162.9
3 308196.2
5 308264.8
3 309404.5
3 310005.2
1 311720.7
4 317743.
5 317748.
3 319719.4
3 321358.8
5 321375.1
7 321398.6
5 321949.1
7 321955.8
9 321964.7
3 322403.1
7 322701.1
3 323024.0
5 323049.4
7 323088.2
```

```
 5 324212.0
 3 327225.7
 1 329633.1
 3 329654.2
 5 329690.9
 5 332690.3
 5 333116.4
 5 333333.4
 7 333358.4
 9 333395.0
 3 337602.9
 5 337616.4
 7 337636.7
 3 339881.
 5 340049.5
 3 340075.8
 1 340090.3
 7 341368.5
 3 343255.7
 5 344181.
 1 345093.9
 7 345444.
 9 346525.1
11 346253.0
 9 346577.5
 5 346656.0
 3 346713.1
 5 347099.5
 7 347101.3
 9 347103.7
 7 348859.5
 3 354796.
 3 357088.
 7 358046.
 9 358638.3
11 358639.0
 9 358688.9
 5 358725.5
 9 358800.
 7 359122.2
 3 363561.
 3 364896.
```

```
15 365585.
5 366027.0
3 369926.
15 370438.
15 373748.
5 376637.
3 381104.8
5 381919.
7 381958.
3 384313.
5 384350.
5 385637.5
5 385816.2
-1 -1
```

B.11   c-3.in

```
520177.8
2 0.0
2 64484.2
4 64591.3
2 302847.9
2 320048.5
4 320080.0
4 324880.2
6 324890.9
2 401346.7
2 408308.9
4 408322.2
4 410333.8
6 410338.2
8 410434.1
2 445366.1
2 448854.
4 448861.
6 449887.4
14 449938.2
18 449948.4
2 468765.
6 470763.
10 471368.
14 471403.0
18 471407.4
22 471407.9
2 482659.
6 483931.
10 484309.
14 484343.8
18 484346.6
22 484346.9
6 492473.
14 492743.
36 492745.
-1 -1
```

B.12 c-4.in

```
3162450.
1 0.
3 2411266.
1 2455165.
3 2455152.
5 2455288.
3 2483240.
15 2857308.
3 2859350.
3 2991680.
3 3053060.
3 3086420.
3 3106750.
3 3118760.
-1 -1
```

B.13   h-0.in

```
109678.758
2 0.000
8 82259
14 97492.3
22 102823.9
32 105291.6
44 106632.15
58 107440.42
74 107965.03
-1 -1
```

B.14   h-1.in

```
100000000.000
1 0.000
-1 -1
```

B.15   n-0.in

```
117345
4   0.000
6 19223
4 19231
6 28840
2 83285.5
4 83319.3
6 83366
2 86131.4
4 86223.2
6 88109.5
4 88153.4
2 88173
2 93582.3
2 94772.2
4 94794.8
6 94832.1
8 94883.1
2 95476.5
4 95494.9
6 95533.2
4 96751.7
4 96788.2
6 96864.2
2 97770.1
4 97805.8
6 99665
4 99658
2 103618.1
4 103668.1
6 103736.8
2 104142.2
4 104227.4
4 104615.4
2 104654.9
4 104665
6 104684
8 104718
10 104767
6 104810.9
8 104882.7
```

```
2 104864
4 104890
6 104957
2 104987
4 104998
6 105011
8 105020
4 105120.8
6 105144.3
2 106478.6
2 106760.5
4 106780.1
6 106816.1
8 106870.7
2 106982.7
4 106998.3
6 107039
4 107447.2
2 109813.5
4 109857.8
6 109927.9
2 110029.2
4 110108.5
4 110196
6 110214
8 110248
10 110304
2 110221
4 110275
6 110288
8 110339
4 110221.7
2 110244.6
6 110311
8 110373
2 110325
4 110351
6 110403
4 110448.3
6 110470.5
4 110521.9
6 110545.8
```

```
2 112294.8
4 112320.8
2 112565.9
4 112610.6
6 112682.6
2 112735
4 112823
4 112751
6 112763
8 112799
10 112862
4 112801
2 112816
6 112820
8 112890.2
6 112825
6 112825
8 112892
2 112855
4 112874
6 112912
4 112929.2
6 112947.5
2 114015
4 114072
6 114146
2 114130
4 114163
28 114160
12 114182
8 114248
4 114193
2 114209
6 114196
8 114275
4 114232.2
6 114290.5
6 114259
6 114274
2 114809
4 114890
6 114942
```

```
6 114950
20 114988
14 115004
-1 -1
```

B.16   n-1.in

```
238846.7
1 0.000
3 49.1
5 131.3
5 15315.7
1 32687.1
5 47167.7
7 92237.9
5 92251.3
3 92252.9
8 109218.2
1 109224.8
5 144189.1
1 148909.37
3 148940.97
5 149077.33
3 149188.74
3 155129.9
3 164611.6
3 166522.48
5 166583.26
7 166679.45
3 166765.7
3 168893.04
1 170573.38
3 170608.63
5 170667
5 174212.93
1 178274.17
5 186512.38
7 186571.8
9 186653.35
5 187092.2
3 187438.34
5 187462.38
7 187492.72
5 188858.09
3 188909.89
1 188937.95
7 189336
3 190121.15
```

```
 1 196541.09
 3 196592.88
 5 196712.17
 3 197859.28
 3 202169.9
 3 202714.94
 5 202765.86
 7 202862.06
 1 203164.7
 3 203188.8
 5 203259.7
 3 203532.8
 5 205350.7
 3 205982.1
 5 206038.1
 7 206108.7
 1 206327.5
 5 209675.3
 7 209739.5
 9 209825.3
 5 209926.92
 3 210239.8
 5 210266.3
 7 210301.9
 5 210705.4
 3 210751.5
 1 210777
 7 211030.9
 5 211033.71
 7 211057.07
 9 211061.03
 7 211104.8
 7 211288.02
 9 211295.65
11 211390.77
 3 211335.5
 9 211402.89
 7 211411.25
 5 211416.2
 3 211487.28
 5 211491.16
 1 211750.2
```

```
3 211780.6
5 211828.8
1 214212.4
3 214258.2
5 214385.3
3 214828
15 220717
12 221070.2
9 221074.3
7 221137.6
7 221227.7
9 221232.7
11 221302.2
9 221312.1
1 224027.1
3 224042.9
5 224072.3
7 224115.4
9 224169.3
3 225987.1
5 226011.2
7 226055.2
5 230223
-1 -1
```

B.17   n-2.in

```
382625.5
2 0.000
4 174.5
2 57192.1
4 57252
6 57333.2
6 101023.8
4 101031.5
2 131003.5
2 145876.1
4 145986.5
4 186802.3
6 203072.3
4 203088.9
2 221302.4
2 230404.5
4 230408.6
2 245665.7
4 245701.7
4 267238.5
6 267244.4
2 287535.6
4 287598.1
6 287713.9
2 297150.2
4 297263.1
2 301088.2
2 309132.6
4 309185.8
2 309662.8
4 309698.3
6 309760.5
8 309856.7
2 311691.3
4 311716.1
4 314224
2 317299.9
4 317343.4
6 317402.3
4 317750.8
6 317781.8
```

14 320287.5
4 320977.4
6 321065.8
2 327056.8
4 330238.4
6 330273.5
8 330325.3
10 330396.7
2 332796.6
4 332810
6 332832
8 332860.3
2 333713.1
4 334542.2
6 334568.9
6 336213.4
4 336268
2 336303.1
6 339744.4
8 339855.7
4 341946.2
6 341947.9
4 342693
2 342763.7
14 342752
18 343116
10 354517
14 354955.7
18 355214
2 368525.6
4 368588.3
6 368704.8
4 373342
6 373376
2 374747.4
4 374805.3
2 376756.6
4 376803.3
6 376863.8
8 376953.3
2 377591
4 377608

-1 -1

139

B.18   n-3.in

```
624851
1 0.000
1 67136.4
3 67199.6
5 67343.8
3 130695
1 175463.5
3 175536.7
5 175661.5
5 188885
1 235370
3 377206
1 388858
3 404521
1 405893.2
3 405909
5 405944
3 419967.8
5 419971.3
7 419979.4
5 429158
1 465223
3 465300.6
5 465463.4
3 473032
3 480880
5 484394
7 484525
3 487542
4 494240
5 494338
5 498315
5 499708
21 499851
9 503625
3 505487
5 505518
7 505561
7 506292
3 507022
15 511384
```

```
5 511440
4 511493
5 514638
5 516631
7 516639
9 516650
3 519414
7 521868
3 550218
15 552731
27 554419
15 574940
5 591043
8 593665
7 593704
-1 -1
```

B.19 o-0.in

```
109836.7
5      0.
3      158.5
1      226.5
5      15867.7
1      33792.4
5      73767.81
3      76794.69
3      86625.35
5      86627.37
7      86631.07
5      88630.84
3      88630.30
1      88631.00
5      95476.43
3      96225.5
9      97420.24
7 97420.37 multiple?
5 97420.50 mult
7 97488.14 mult
3 99092.64
5 99093.31
7 99094.52
5 99680.4
7 101135.04
5 101147.21
3 101155.10
5 102116.21
3 102411.65
5 102661.63
9 102865.09 multiple?
7 102908.14 multiple?
5 103869.4 mult
5 105019.0
3 105164.90
9 105385.3 multiple?
7 105408.58 mult
5 105911.3
5 106545.1
3 106627.9
9 106751.2
```

```
7 106765.8
5 107445.4
3 107497.1
9 107573.1
7 107582.7
5 108021.4
3 108057.6
9 108105.7
7 108116.6
5 108412.0
3 108436.1
9 108470.2
7 108477.8
5 108688.4
3 108707.3
9 108731.5
7 108734.4
-1 -1
```

B.20   o-1.in

```
283550.9
4 0.
6 26808.4
4 26829.4
4 40466.9
2 40468.4
6 119837.7
4 120001.1
2 120083.5
6 165987.7
4 165996.0
2 185235.36
4 185340.68
6 185499.20
2 188888.38
4 189068.37
2 195710.4
2 203942.21
2 206730.80
4 206786.34
6 206877.90
8 207002.52
6 206971.3
＜ 206972.3
2 208346.17
4 208392.27
6 208484.24
4 211521.98
6 211712.66
4 212161.94
4 212593.2
2 212762.4
2 214169.74
4 214229.48
2 226851.
6 228723.3
8 228746.9
6 229946.6
4 229968.2
4 231296.05
6 231350.08
```

```
8 231427.99
10 231530.26
6 232462.83
4 232536.06
2 232602.57
2 232480.1
4 232526.7
2 232711.70
4 232745.98
6 232747.51
8 232753.86
6 232796.27
8 232959.26
4 233430.10
2 233544.09
4 234402.48
6 234454.45
2 238626.32
4 238731.54
6 238892.96
2 240328.75
4 240516.28
6 245395.5
2 245767.80
4 245816.29
6 245902.85
8 246028.95
4 248009.1
6 248185.3
2 248425.35
4 248514.23
6 250251.
8 251220.9
6 251224.1
10 252607.7
8 252608.9
4 253046.23
6 253048.35
2 253789.51
4 253791.87
8 254481.5 multiple?
10 254590.7
```

10 254895.2 multiple?
4 254982.2
6 255104.6
4 255140.9
2 255162.6
4 255172.5
2 255281.4
6 255301.3
8 255465.2
2 255622.4
6 255689.6
4 255812.2
8 255691.4
6 255813.1
4 255913.
2 255912.0
6 255755.8
8 255759.4
10 255827.6
12 255977.5
8 255829.4
10 255983.6
4 255843.1
6 255897.2
4 256083.5
6 256087.6
8 256123.1
10 256136.2
6 256125.8
8 256143.3
2 257693.7
4 257797.9
6 257963.8
2 258408.6
4 258601.7
6 259286.2
4 259287.0
4 260959.
6 261042.
8 261180.
4 261261.7
6 261354.3

```
4 261697.5
6 261869.4
10 265220.3
6 265431.5
6 265468.2
8 265578.
8 265639.
6 265705.
4 265762.
2 265859.
6 265665.
8 265691.
10 265761.
12 265925.
8 265763.0
10 265930.2
6 265856.
4 265928.
6 265961.
8 265985.
10 265999.
6 265988.
8 265999.
4 267763.39
6 267770.85
8 267783.40
6 274739.2
8 274782.4
10 274920.
6 275611.
18 275841.3
14 275879.6
10 275951.
2 275997.
10 276066.3
22 276109.1
6 276263.9
10 278140
-1 -1
```

B.21   o-2.in

```
443193.5
1 0.
3 113.4
5 306.8
5 20271.0
1 43183.5
5 60312.1
7 120025.4
5 120052.6
3 120058.5
5 142381.7
3 142382.8
1 142396.9
5 187049.4
3 197086.7
3 210458.5
1 267257.29
3 267375.65
5 267632.59
3 273080.07
5 283758.9
3 283976.6
1 284073.3
3 290956.62
3 293865.26
5 294001.60
7 294221.65
3 297557.50
5 298289.4
1 300228.21
3 300310.31
5 300440.85
5 306584.8
1 313801.07
5 324462.46
7 324658.25
9 324836.41
5 324734.22
3 327227.94
5 327277.18
7 327350.90
```

```
5 329467.98
3 329581.98
1 329643.43
7 331820.2
3 332777.1
3 338565.87
5 338690.34
7 338851.50
1 343302.6
1 350026.1
3 350122.9
5 350302.3
1 356732.
3 356838.
5 357111.
3 358667.4
3 363266.8
1 365515.76
3 365550.60
5 365619.12
7 365719.16
9 365846.46
3 365723.9
3 366486.91
5 366594.01
7 366801.04
3 367952.20
3 368526.37
5 368583.63
7 368684.75
1 370326.7
3 370415.7
5 370524.2
5 370900.6
1 373046.2
3 374575.
5 374662.5
7 374798.6
5 376067.66
5 377375.
5 377687.
5 378408.5
```

```
3 378420.9
1 378438.1
3 379232.
5 379293.
7 379356.
5 380706.
7 380782.
3 381086.
5 392221.
3 392778.
3 394090.
5 394126.
7 394195.
3 394516.45
5 394555.15
7 394612.70
9 394688.44
11 394780.47
1 398135.0
3 398131.4
5 398127.3
7 398137.4
9 398218.8
7 398474.3
5 398544.3
3 398582.8
5 400354.8
3 400464.7
1 400518.4
5 401379.
7 401475.4
9 401609.1
5 401530.
5 401787.
7 402530.
7 403374.
3 403526.
3 405805.1
5 405834.1
7 405833.0
5 414675.
7 415181.
```

```
7 422977.
7 424998.
5 426338.
3 428487.
5 428606.
7 428769.
3 430025.
3 437015.0
3 438241.0
5 438303.2
7 438395.2
9 438517.5
3 439278.1
5 439329.5
7 439427.6
7 442710.
-1 -1
```

B.22   o-3.in

```
624396.5
2 0.0
4 386.5
2 71177.0
4 71308.4
6 71492.9
6 126936.3
4 126950.3
2 164366.9
2 180481.3
4 180724.6
4 231275.1
6 255156.7
4 255186.0
2 289016.1
4 289024.0
2 357614.8
2 390161.1
4 390248.2
4 419533.5
6 419550.2
2 438588.5
4 438723.6
6 438970.5
2 452808.0
4 453073.0
2 467231.1
4 467346.5
2 468075.4
4 468154.2
6 468289.7
8 468499.4
4 474217.8
2 478587.7
4 478682.2
6 478811.3
4 482667.5
6 482923.1
2 485823.1
2 492880.
4 494907.5
```

6 494986.3
8 495098.7
10 495252.8
2 499506.4
4 499535.3
6 499582.0
8 499646.6
4 501511.3
6 501566.4
6 503834.5
4 503947.9
2 504021.7
4 510560.
6 510567.
6 510746.1
8 510978.5
4 514217.
2 514368.
2 518684.
4 518690.
2 539368.
4 547311.
6 547336.
2 549792.
4 549855.
6 552034.
14 552490
2 554461
2 568638.
4 568773.
6 569020.
8 570791.
2 573696.
4 573907.
6 574373.
2 575204.
4 575373.
4 575819.
6 575853.
2 576591.
4 576735.
6 576947.

```
2 581721.
4 581743.
4 584552.
6 584768.
14 587850.
2 590071.
8 591767.
6 592999.
4 593627.
6 593708.
6 594007.
8 594080.
4 594337.
6 594542.
6 596299.
8 596477.
2 597254.
14 597352.
4 597726.
2 597863.
4 600092.
6 600106.
8 602977.
6 602434.
6 615431.
4 615460.
4 616588.
-1 -1
```

B.23 o-4.in

```
918702.
1 0.
1 82121.2
3 82257.9
5 82564.1
3 158798.
1 213641.7
3 213797.4
5 214066.2
5 231722.
1 287909.
3 547150.0
1 561278.
3 580826.
1 582983.6
3 583019.9
5 583097.2
3 600925.5
5 600936.6
7 600956.1
5 612617.
1 653099.7
3 653262.2
5 653605.0
3 664486.
3 672695.
3 677333.
5 677532.
7 677847.
3 684124.
1 689585.6
3 689699.6
5 689890.3
5 694646.
5 697170.
3 704360.
5 704424.
7 704527.
1 707630.
5 708154.
3 708296.
```

1 708379.
7 712967
3 709277.
3 722666.
1 731667.
3 736108.
5 736126.
3 737883.
3 742401.
5 742407.
7 742421.
5 746280.
7 749857.
3 796263.
3 802452.
7 806625.
5 808351.
3 824280.
3 829588.
3 831047.
5 831213.
7 831504.
3 832251.
3 835151.
5 835321.
5 837834.
5 837864.
3 839616.
7 840832.
7 841220.
3 841280.
5 841374.
7 841497.
5 842105.
5 843290.
3 843397.
1 843449.
7 847129.
3 847465.
3 860874.
7 861975.
5 862419.

```
3 874447.
7 875365.
3 898580.
7 899671.
5 901344.
5 902442.
5 902592.
7 904497.
7 906404.
-1 -1
```